

Architecture-aware Coding for Distributed Storage: Repairable Block Failure Resilient Codes ¹

Gokhan Calis and O. Ozan Koyluoglu

Abstract

In large scale distributed storage systems (DSS) deployed in cloud computing, correlated failures resulting in simultaneous failure (or, unavailability) of blocks of nodes are common. In such scenarios, the stored data or a content of a failed node can only be reconstructed from the available live nodes belonging to the available blocks. To analyze the resilience of the system against such block failures, this work introduces the framework of Block Failure Resilient (BFR) codes, wherein the data (e.g., a file in DSS) can be decoded by reading out from a same number of codeword symbols (nodes) from a subset of available blocks of the underlying codeword. Further, repairable BFR codes are introduced, wherein any codeword symbol in a failed block can be repaired by contacting a subset of remaining blocks in the system. File size bounds for repairable BFR codes are derived, and the trade-off between per node storage and repair bandwidth is analyzed, and the corresponding minimum storage regenerating (BFR-MSR) and minimum bandwidth regenerating (BFR-MBR) points are derived. Explicit codes achieving the two operating points for a special case of parameters are constructed, wherein the underlying regenerating codewords are distributed to BFR codeword symbols according to combinatorial designs. Finally, BFR locally repairable codes (BFR-LRC) are introduced, an upper bound on the resilience is derived and optimal code construction are provided by a concatenation of Gabidulin and MDS codes. Repair efficiency of BFR-LRC is further studied via the use of BFR-MSR/MBR codes as local codes. Code constructions achieving optimal resilience for BFR-MSR/MBR-LRCs are provided for certain parameter regimes. Overall, this work introduces the framework of block failures along with optimal code constructions, and the study of architecture-aware coding for distributed storage systems.

I. INTRODUCTION

A. Background

Increasing demand for storing and analyzing *big-data* as well as several applications of cloud computing systems require efficient cloud computing infrastructures. Under today's circumstances where the data is growing exponentially, it is crucial to have storage systems that guarantee no permanent loss of the data. However, one inevitable nature of the storage systems

The authors are with Department of Electrical and Computer Engineering, The University of Arizona. Email: {gcalis, ozan}@email.arizona.edu. This paper was in part presented at 2014 IEEE International Symposium on Information Theory (ISIT 2014), Honolulu, HI, June 2014.

This work is supported in part by the National Science Foundation under Grants No CCF-1563622 and CNS-1617335.

is node failures. In order to provide resilience against such failures, redundancy is introduced in the storage.

Classical redundancy schemes range from *replication* to *erasure coding*. Erasure coding allows for better performance in terms of reliability and redundancy compared to replication. To increase repair bandwidth efficiency of erasure coded systems, regenerating codes are proposed in the seminal work of Dimakis et al. [1]. In such a model of distributed storage systems (DSS), the file of size \mathcal{M} is encoded to n nodes such that any $k \leq n$ nodes (each with α symbols) allow for reconstructing the file and any $d \geq k$ nodes (with $\beta \leq \alpha$ symbols from each) reconstruct a failed node with a repair bandwidth $\gamma = d\beta$. The trade-off between per node storage (α) and repair bandwidth (γ) is characterized and two ends of the trade-off curve are named as minimum storage regenerating (MSR) and minimum bandwidth regenerating (MBR) points [1]. Several explicit codes have been proposed to achieve these operating points recently [2]–[8].

Another metric for an efficient repair is repair degree d and regenerating codes necessarily have $d \geq k$. Codes with locality and locally repairable codes with regeneration properties [9]–[18] allow for a small repair degree, wherein a failed node is reconstructed via local connections. Instances of such codes are recently considered in DSS [19], [20]. Small repair degree has its benefits in terms of the implementation of DSS since a failed node requires only *local* connections. In particular, when more nodes are busy for recovery operations, this in turn creates additional access cost in the network.

In large-scale distributed storage systems (such as GFS [21]), *correlated failures* are unavoidable. As analyzed in [22], these simultaneous failures of multiple nodes affect the performance of computing systems severely. The analysis in [22] further shows that these correlated failures arise due to the *failure domains*, e.g., nodes connected to the same power source, the same update group, or the same cluster (e.g., rack) exhibit these *structured failure bursts*. The unavailability periods are transient, and largest failure bursts almost always have significant rack-correlation. For example, in Fig. 1 there are three racks in a DSS, each connected to the same switch. Assume that top-of-rack (TOR) switch of the first rack is failed; hence, all the disks in the rack are assumed to be failed or unavailable (same failure domain). Now consider that, while the first rack is unavailable, a user asks for some data D stored in one of the disks in the first rack. If both the data and its corresponding redundancy were stored all together in the first rack, then the user would not be able to access the data until the TOR switch works properly. On the other hand, assume that redundancy is distributed to the other two racks; then, the user could connect to those two racks in order to reconstruct the data. Furthermore, if the disk storing the data fails in the first rack, then the repair process could be performed similarly since the failed disk could connect to other racks to download some amount of data for repair process. This architecture is also relevant to disk storage, where the disk is divided into sectors, each can be unavailable or under failure. To overcome from failures having such patterns, a different approach is needed.

In terms of data recovery operations, such an architecture can be considered as a relaxation of regenerating coded system. In particular, considering a load balancing property for the racks

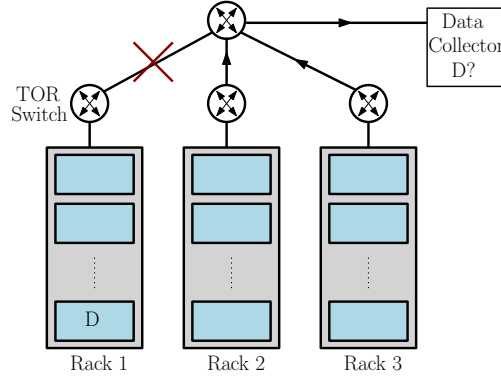


Fig. 1: A data-center architecture where top-of-rack (TOR) switch of the first rack fails

utilized in the recovery operations, we focus on the following model. Regenerating codes allow the failed node to connect to any d nodes for repair purposes whereas we consider failed node connecting to a total of d nodes in some restricted subset of nodes, i.e., any $\frac{d}{2}$ nodes in the second and third racks respectively for the example in Fig. 1. Similarly, any k property of regenerating codes for data-reconstruction is also relaxed, i.e., DC can connect to $\frac{k}{3}$ nodes in each rack in the example above. The outcome of such a relaxation is directly related to the performance of the DSS in terms of per node storage and repair bandwidth trade-off. For example, consider a DSS where a file of size \mathcal{M} is stored over $n = 10$ nodes such that any $k = 4$ nodes are enough to reconstruct the data. In addition, any $d = 4$ nodes are required to regenerate a failed node. For such a system that uses regenerating code, the trade-off curve can be obtained as in Fig. 2(c). Now consider that, $n = 10$ nodes are distributed into two distinct groups such that each *block* has $\frac{n}{2} = 5$ nodes. Note that, a failed node in one of the blocks can now be repaired by connecting to any $d = 4$ nodes in the other block. Also, DC can contact $\frac{k}{2} = 2$ nodes per block to reconstruct the original message \mathcal{M} . For such a relaxation, we can obtain the corresponding trade-off curve between per node storage and repair bandwidth as in Fig. 2(c). Observe that the new MSR point, called *BFR-MSR*, has significantly lower repair bandwidth than MSR point as a result of this relaxation. In this paper, we further show that the gap between the repair bandwidth of BFR-MSR and MBR point can be made arbitrarily small for large systems while keeping the per node storage of BFR-MSR point same as MSR point. Therefore, such a relaxation of regenerating codes allow for simultaneous achievability of per-node storage of MSR point and repair bandwidth of MBR point.

B. Contributions and organization

The contributions of this paper can be summarized as follows:

- We develop a framework to analyze resilience against block failures in DSS with node repair efficiencies. We consider a DSS with a single failure domain, where nodes belonging to the same failure group form a block of the codeword.
- We introduce block failure resilient (BFR) codes, which allow for data collection from any $b_c = b - \rho$ blocks, where b is the number of blocks in the system and ρ is the resilience

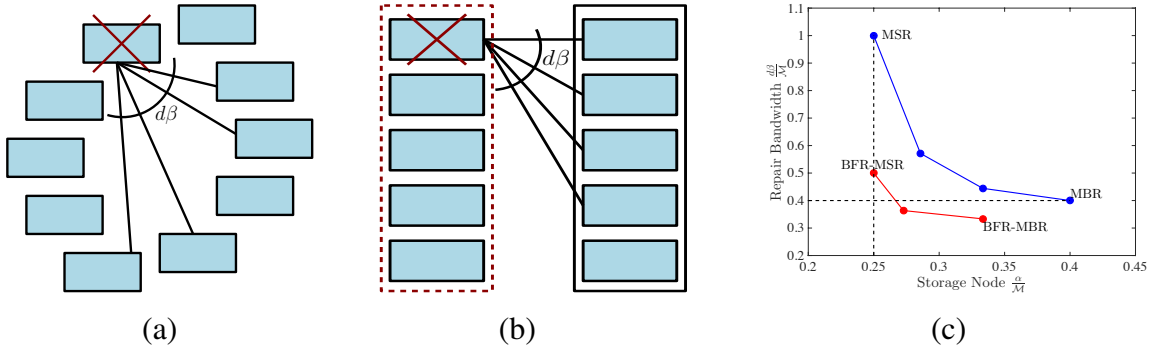


Fig. 2: (a) Node repair in regenerating codes. (b) Node repair in relaxation of regenerating codes. (c) Trade-off curves in toy example.

parameter of the code. Considering a load-balancing among blocks, a same number of nodes are contacted within these b_c blocks. (A total of $k = k_c b_c$ nodes and downloading α , i.e., all-symbols from each.) This constitutes data collection property of BFR codes. ($\rho = 0$ case can be considered as a special case of batch codes introduced in [23].) We emphasize that our results in this part of the manuscript also consider the case of $\rho > 0$. Furthermore, we introduce repairability in BFR codes, where any node of a failed block can be reconstructed from any d_r nodes of any remaining $b_r = b - \sigma \leq b - 1$ blocks. (A total of $d = d_r b_r$ nodes and downloading β symbols from each.) As introduced in [1], we utilize graph expansion of DSS employing these repairable codes, and derive file size bounds and characterize BFR-MSR and BFR-MBR points for a wide set of parameter regimes. We emphasize the relation between d_r and k_c as well as ρ and σ to differentiate all cases.¹

- The main focus of this manuscript is on code constructions for $\sigma > 0$ and $\rho = 0$ scenario, where the data access assumes the availability of all ρ blocks and partial data access (“repair”) is over any given subset of blocks of size $b - \sigma$.
- We construct explicit codes achieving these points for a set of parameters when $d_r \geq k_c$, $\rho = 0$ and $\sigma = 1$. For a system with $b = 2$ blocks case, we show that achieving both MSR and MBR properties simultaneously is asymptotically possible. (This is somewhat similar to the property of Twin codes [24], but here the data collection property is different.) Then, for a system with $b \geq 3$ blocks case, we consider utilizing multiple codewords, which are placed into DSS via a combinatorial design (projective planes) based codeword placement.
- We also provide the code constructions for any $\sigma < b - 1$ and $\rho = 0$ by utilizing Duplicated Combination Block Designs (DCBD). These codes can achieve BFR-MSR and BFR-MBR points for a wider set of parameters than the codes described above.
- We introduce BFR locally repairable codes (BFR-LRC). We establish an upper bound on the resilience of such codes (using techniques similar to [12]) and propose resilience-optimal

¹The case with $\frac{d}{b-\sigma} \geq \frac{k}{b-\rho}$ and $\sigma > \rho$ remains open in terms of a general proof for the minimum possible file size bound. However, the obtained bound is shown to be tight by code constructions for certain parameter regimes.

constructions that use a two step encoding process and combinatorial design placement.

- We also analyze the case where regenerating codes are used to improve repair efficiency of BFR-LRC codes. These codes are called BFR-MSR/MBR-LRC and they have a better performance in terms of repair bandwidth. We identify the upper bound on the file size that can be stored for both cases and also propose a construction that utilizes DCBD based construction to achieve the studied bound for certain parameter regimes.
- We provide codes with table-based type of relaxation for repairable BFR in order to operate within wider set of parameters.

The rest of the paper is organized as follows. In Section II, we introduce BFR codes and provide preliminaries. Section III focuses on file size bounds for efficient repair where we study BFR-MSR and BFR-MBR points. In Section IV, we provide explicit code constructions. We discuss BFR-LRC and local regeneration in BFR-LRC in Section V. We extend our discussion in Section VI where we analyze repair time of DSS with BFR as well as other codes, e.g., regenerating codes, and also propose a relaxation for the BFR model, where we provide explicit codes achieving the performance of BFR-MSR/MBR. Finally we conclude in Section VII.

C. Related Work

In the seminal work of Dimakis et al. [1], *regenerating codes* are presented, which are proposed to improve upon classical erasure codes in terms of repair bandwidth. The authors focus on a setting similar to maximum distance separable (MDS) codes as regenerating codes also have *any k out of n* property which allows data collector to connect to any k nodes to decode data stored over n nodes; however, they show that by allowing a failed node to connect to $d \geq k$ nodes, they can significantly reduce the repair bandwidth. Thus, a trade-off between per node storage and repair bandwidth for a single node repair is presented. Such a repair process is referred to as *functional* repair since the *newcomer* node may not store the same data as the failed node; however in terms of functionality, they are equivalent. In [1], [6], [7], functional repair is studied to construct codes achieving the optimality at the two ends of the trade-off curve. In [3], the focus is on *exact* repair where the failed node is repaired such that the newcomer stores exactly the same data as the failed node stored. The proposed code construction is optimal for all parameters (n, k, d) for minimum bandwidth regeneration (MBR) point as well as optimal for $(n, k, d \geq 2k - 2)$ for minimum storage regeneration (MSR) point. It can be deduced that since $n - 1 \geq d$ necessarily, these codes' rate is bounded by $\frac{1}{2} + \frac{1}{2n}$ for MSR point. [4] utilizes two parity nodes to construct exact regeneration codes that are optimal in repair bandwidth, whereas [2], [8] focus on exact-repair for systematic node repair through permutation-matrix based constructions. Our BFR model here can be seen as a structured relaxation of the data collection and repair process of regenerating codes as discussed in the previous subsection.

Combinatorial designs are utilized in constructing fractional repetition codes in [25]. In this construction, MDS codewords are replicated and these replicated symbols are placed to storage nodes according to the combinatorial designs. The resulting coding scheme can be viewed as the

relaxation of regenerating codes where the repairs are performed in a *table-based* manner (instead of “any d ” property). Other works that utilize combinatorial designs include [26]–[28]. For the constructions proposed in this paper, we consider having (in certain cases precoded versions of) *regenerating codewords* placed to storage nodes according to combinatorial designs. As compared to fractional repetition codes, this allows to have bandwidth efficient repair with “any d_r ” property in the block failure context studied here. In the last part of the sequel, we revisit and provide codes with a table-based type of relaxation for repairable BFR model, where instead of “any d_r ” per helper block we consider “any d ” per helper sub-block.

Proposed recently in [11], *local* codes operate node repairs within some local group thereby reducing repair degree. In [9], minimum distance of locally repairable codes are studied for scalar case only and Pyramid codes are shown to be optimal in terms of minimum distance [14]. [13] generalizes the bound by allowing multiple local parities per local group for the scalar linear code. Next, [10] focuses on the minimum distance bound for vector codes which have only one local parity per group as well as characterize the trade-off between per node storage and resilience. In [12], [29], the authors generalize the bound on minimum distance for vectors codes which can have multiple parity nodes per local group and characterize the trade-off between per node storage and resilience. [30] proposes locally repairable codes with small field sizes by considering polynomial construction of Reed-Solomon codes. For LRCs, when the number of failures exceed the corresponding minimum distance of the local group, the local recovery would not be possible. Our BFR model here can be considered as the opposite of locality property, where non-local nodes have to be contacted for recovery operations.

Recently, another type of erasure codes called Partial-MDS are proposed for RAID systems to tolerate not just disk failures but also sector failures [31]. Considering a stripe on a disk system as an $r \times n$ array where disks are represented by columns and all disks have r sectors, PMDS codes tolerate an erasure of any m sectors per row plus any s additional elements. Later, relaxation of PMDS codes are studied in [32] which allow erasure of any m columns plus any s additional elements. These codes are called Sector-Disk (SD) codes. Our BFR model can also be considered as a class of SD codes since BFR construction tolerates ρ block failures which can be treated as disks (columns) in SD code construction. However, for the full access case, BFR codes do not have any additional erasure protection beyond $m = \rho$ disk failures. And, for the partial access case, BFR codes allow additional erasure protection but the code tolerates any additional $c - k_c$ number of erasures per block (disk) rather than s number of erasures that can occur anywhere over the remaining disks. On the other hand, repair process in SD codes utilize the row-wise parities (one sector from other disks) to correct the corresponding sectors in the failed disks, but repairable BFR allows the failed node (sector) in a block (disk) to contact any set of nodes in other blocks.

We note that the blocks in our model can be used to model clusters (e.g., racks) in DSS. Such a model is related to the work in [33] which differentiates between within-rack communication and cross-rack communication. Our focus here would correspond to the case where within the

failure group, communication cost is much higher than the cross-group communication cost,⁷ as no nodes from the failed/unavailable group can be contacted to regenerate a node. Another recent work [34] considers clustered storage as well, but the repair mechanism is different than our model, where, similar to [33], [34] considers availability of nodes in the block that has a failed node, i.e., there is no consideration of failure of blocks.

II. BACKGROUND AND PRELIMINARIES

A. Block failure resilient codes and repairability

Consider a code \mathcal{C} which maps \mathcal{M} symbols (over \mathbb{F}_q) in \mathbf{f} (file) to length n codewords (nodes) $\mathbf{c} = (c_1, \dots, c_n)$ with $c_i \in \mathbb{F}_q^\alpha$ for $i = 1, \dots, n$. These codewords are distributed into b blocks each with block capacity $c = \frac{n}{b}$ nodes per block. We have the following definition.

Definition 1 (Block Failure Resilient (BFR) Codes). *An $(n, b, \mathcal{M}, k, \rho, \alpha)$ block failure resilient (BFR) code encodes \mathcal{M} elements in \mathbb{F}_q (\mathbf{f}) to n codeword symbols (each in \mathbb{F}_q^α) that are grouped into b blocks such that \mathbf{f} can be decoded by accessing to any $\frac{k}{b-\rho}$ nodes from each of the $b - \rho$ blocks.*

We remark that, in the above, ρ represents the resilience parameter of the BFR code, i.e., the code can tolerate ρ block erasures. Due to this data collection (file decoding) property of the code, we denote the number of blocks accessed as $b_c = b - \rho$ and number of nodes accessed per block as $k_c = \frac{k}{b_c}$. Noting that $k_c \leq c$ should be satisfied, we differentiate between *partial* block access, $k_c < c$, and *full* block access $k_c = c$. Throughout the paper, we assume $b \mid n$. i.e., c is integer, and $(b - \rho) \mid k$, i.e., k_c is integer.

Remarkably, any MDS array code [35] can be utilized as BFR codes for the full access case. In fact, such an approach will be optimal in terms of minimum distance, and therefore for resilience ρ . However, for $k_c < c$, MDS array codes may not result in an optimal code in terms of the trade-off between resilience ρ and code rate $\frac{\mathcal{M}}{n\alpha}$. Concatenation of Gabidulin codes and MDS codes as originally proposed in [12] gives optimal BFR codes for all parameters. For completeness, we provide this coding technique adapted to generate BFR codes in Appendix A. We remark that this concatenation approach is used for locally repairable codes in [12], for locally repairable codes with minimum bandwidth node repairs in [16], thwarting adversarial errors in [36], [37], cooperative regenerating codes with built-in security mechanisms against node capture attacks in [38] and for constructing PMDS codes in [39]. In this work, we focus on repairable BFR codes, as defined in the following.

Definition 2 (Block Failure Resilient Regenerating Codes (BFR-RC)). *An $(n, b, \mathcal{M}, k, \rho, \alpha, d, \sigma, \beta)$ block failure resilient regenerating code (BFR-RC) is an $(n, b, \mathcal{M}, k, \rho, \alpha)$ BFR code (data collection property) with the following repair property: Any node of a failed block can be reconstructed by accessing to any $d_r = \frac{d}{b-\sigma}$ nodes of any $b_r = b - \sigma$ blocks and downloading β symbols from each of these $d = b_r d_r$ nodes.*

We assume $(b - \sigma) \mid d$, i.e., d_r is integer. (Note that d_r should necessarily satisfy $\frac{d}{b-\sigma} = \frac{8}{d_r} \leq c = \frac{n}{b}$ in our model.) We consider the trade-off between the *repair bandwidth* $\gamma = d\beta$ and *per node storage* α similar to the seminal work of Dimakis et al. [1]. In particular, we define $\alpha_{\text{BFR-MSR}} = \frac{M}{k}$ as the minimum per node storage and $\gamma_{\text{BFR-MBR}} = \alpha_{\text{BFR-MBR}}$ as the minimum repair bandwidth for an $(n, b, \mathcal{M}, k, \rho, \alpha, d, \sigma, \beta)$ BFR-RC.

Remark 3. Having $\sigma > 0$ allows partial data reconstruction as one can regenerate the unavailable/failed node inside a block.

B. Information flow graph

The operation of a DSS employing such codes can be modeled by a multicasting scenario over an information flow graph [1], which has three types of nodes: 1) Source node (S): Contains original file f . 2) Storage nodes, each represented as x_i with two sub-nodes $((x_i^{\text{in}}, x_i^{\text{out}}))$, where x_i^{in} is the sub-node having the connections from the live nodes, and x_i^{out} is the storage sub-node, which stores the data and is contacted for node repair or data collection (edges between each x_i^{in} and x_i^{out} has α -link capacity). 3) Data collector (DC) which contacts x_i^{out} sub-nodes of k live nodes (with edges each having ∞ -link capacity). (As described above, for BFR codes these k nodes can be any $\frac{k}{b-\rho}$ nodes from each of the $b - \rho$ blocks.) Then, for a given graph \mathcal{G} and DCs DC_i , the file size can be bounded using the max flow-min cut theorem for multicasting utilized in network coding [1], [40].

Lemma 4 (Max flow-min cut theorem for multicasting).

$$\mathcal{M} \leq \min_{\mathcal{G}} \min_{\text{DC}_i} \text{maxflow}(S \rightarrow \text{DC}_i, \mathcal{G}),$$

where $\text{flow}(S \rightarrow \text{DC}_i, \mathcal{G})$ represents the flow from the source node S to DC_i over the graph \mathcal{G} .

Therefore, \mathcal{M} symbol long file can be delivered to a DC, only if the min cut is at least \mathcal{M} . In the next section, similar to [1], we consider k successive node failures and evaluate the min-cut over possible graphs, and obtain file size bounds for DSS operating with BFR-RC.

C. Vector codes

An $(n, M, d_{\min}, \alpha)_q$ vector code $\mathcal{C} \subseteq (\mathbb{F}_q^\alpha)^n$ is a collection of M vectors of length $n\alpha$ over \mathbb{F}_q . A codeword $\mathbf{c} \in \mathcal{C}$ consists of n blocks, each of size α over \mathbb{F}_q . We can replace each α -long block with an element in \mathbb{F}_q^α to obtain a vector $\mathbf{c} = (\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_n) \in \mathbb{F}_q^{n\alpha}$. The minimum distance, d_{\min} , of \mathcal{C} is defined as minimum Hamming distance between any two codewords in \mathcal{C} .

Definition 5. Let \mathbf{c} be a codeword in \mathcal{C} selected uniformly at random from M codewords. The minimum distance of \mathcal{C} is defined as

$$d_{\min} = n - \max_{\mathcal{A} \subseteq [n]: H(\mathbf{c}_{\mathcal{A}}) < \log_q M} |\mathcal{A}|,$$

where $\mathcal{A} = \{i_1, \dots, i_{|\mathcal{A}|}\} \subseteq [n]$, $\mathbf{c}_{\mathcal{A}} = (\mathbf{c}_{i_1}, \dots, \mathbf{c}_{i_{|\mathcal{A}|}})$, and $H(\cdot)$ denotes q -entropy.

A vector code is said to be maximum distance separable (MDS) code if $\alpha \mid \log_q M$ and $d_{\min} = n - \frac{\log_q M}{\alpha} + 1$. A linear $(n, M, d_{\min}, \alpha)_q$ vector code is a linear subspace of $\mathbb{F}_q^{\alpha n}$ of dimension $\mathcal{M} = \log_q M$. An $[n, \mathcal{M}, d_{\min}, \alpha]_q$ array code is called *MDS array code* if $\alpha \mid \mathcal{M}$ and $d_{\min} = n - \frac{\mathcal{M}}{\alpha} + 1$.

The encoding process of an $(n, M = q^{\mathcal{M}}, d_{\min}, \alpha)_q$ vector code can be summarized by $\mathbb{G} : \mathbb{F}_q^{\mathcal{M}} \rightarrow (\mathbb{F}_q^{\alpha})^n$. The encoding function is defined by an $\mathcal{M} \times n\alpha$ generator matrix $G = [\mathbf{g}_1^1, \dots, \mathbf{g}_1^{\alpha} \mid \dots \mid \mathbf{g}_n^1, \dots, \mathbf{g}_n^{\alpha}]$ over \mathbb{F}_q .

D. Maximum rank distance codes

Gabidulin codes [41], are an example of class of rank-metric codes, called maximum rank distance (MRD) codes. (These codes will be utilized later in the sequel.)

Let \mathbb{F}_{q^m} be an extension field of \mathbb{F}_q . An element $v \in \mathbb{F}_{q^m}$ can be represented as the vector $\mathbf{v} = (v_1, \dots, v_m)^T \in \mathbb{F}_q^m$, such that $v = \sum_{i=1}^m v_i b_i$, for a fixed basis $\{b_1, \dots, b_m\}$ of the extension field \mathbb{F}_{q^m} . Using this, a vector $\mathbf{v} = (v_1, \dots, v_N) \in \mathbb{F}_{q^m}^N$ can be represented by an $m \times N$ matrix $\mathbf{V} = [v_{i,j}]$ over \mathbb{F}_q , which is obtained by replacing each v_i of \mathbf{v} by its vector representation $(v_{i,1}, \dots, v_{i,m})^T$.

Definition 6. The rank of a vector $\mathbf{v} \in \mathbb{F}_{q^m}^N$, $\text{rank}(\mathbf{v})$, is defined as the rank of its $m \times N$ matrix representation \mathbf{V} (over \mathbb{F}_q). Then, rank distance is given by

$$d_R(\mathbf{v}, \mathbf{u}) = \text{rank}(\mathbf{V} - \mathbf{U}).$$

An $[N, K, D]_{q^m}$ rank-metric code $C \subseteq \mathbb{F}_{q^m}^N$ is a linear block code over \mathbb{F}_{q^m} of length N with dimension K and minimum rank distance D . A rank-metric code that attains the Singleton bound $D \leq N - K + 1$ in rank-metric is called *maximum rank distance* (MRD) code. Gabidulin codes can be described by evaluation of *linearized polynomials*.

Definition 7. A linearized polynomial $f(y)$ over \mathbb{F}_{q^m} of q -degree t has the form

$$f(y) = \sum_{i=0}^t a_i y^{q^i}$$

where $a_i \in \mathbb{F}_{q^m}$, and $a_t \neq 0$.

Process of encoding a message (f_1, f_2, \dots, f_K) to a codeword of an $[N, K, D = N - K + 1]$ Gabidulin code over \mathbb{F}_{q^m} has two steps:

- Step 1: Construct a data polynomial $f(y) = \sum_{i=1}^K f_i y^{q^{i-1}}$ over \mathbb{F}_{q^m} .
- Step 2: Evaluate $f(y)$ at $\{y_1, y_2, \dots, y_n\}$ where each $y_i \in \mathbb{F}_{q^m}$, to obtain a codeword $\mathbf{c} = (f(y_1), \dots, f(y_N)) \in \mathbb{F}_{q^m}^N$.

Remark 8. For any $a, b \in \mathbb{F}_q$ and $v_1, v_2 \in \mathbb{F}_{q^m}$, we have $f(av_1 + bv_2) = af(v_1) + bf(v_2)$.

Remark 9. Given evaluations of $f(\cdot)$ at any K linearly independent (over \mathbb{F}_q) points in \mathbb{F}_{q^m} , one can reconstruct the message vector. Therefore, an $[N, K, D]$ Gabidulin code is an MDS code and can correct any $D - 1 = N - K$ erasures.

E. Locally repairable codes

Recently introduced locally repairable codes reduce repair degree by recovering a symbol via contacting small number of helper nodes for repair.

Definition 10 (Punctured Vector Codes). *Given an $(n, M, d_{\min}, \alpha)_q$ vector code \mathcal{C} and a set $\mathcal{S} \subset [n]$, $\mathcal{C}|_{\mathcal{S}}$ is used to denote the code obtained by puncturing \mathcal{C} on $[n] \setminus \mathcal{S}$. In other words, codewords of $\mathcal{C}|_{\mathcal{S}}$ are obtained by keeping those vector symbols in $\mathbf{c} = (\mathbf{c}_1, \dots, \mathbf{c}_n) \in \mathcal{C}$ which have their indices in set \mathcal{S} .*

Definition 11. *An $(n, M, d_{\min}, \alpha)_q$ vector code \mathcal{C} is said to have (r, δ) locality if for each vector symbol $\mathbf{c}_i \in \mathbb{F}_q^n$, $i \in [n]$, of codeword $\mathbf{c} = (\mathbf{c}_1, \dots, \mathbf{c}_n) \in \mathcal{C}$, there exists a set of indices $\Gamma(i)$ such that*

- $i \in \Gamma(i)$
- $|\Gamma(i)| \leq r + \delta - 1$
- Minimum distance of $\mathcal{C}|_{\Gamma(i)}$ is at least δ .

Remark 12. *The last requirement in Definition 11 implies that each element $j \in \Gamma(i)$ can be written as a function of any set of r elements in $\Gamma(i) \setminus \{j\}$.*

III. FILE SIZE BOUND FOR REPAIRABLE BFR CODES

In this section, we perform an analysis to obtain file size bounds for repairable BFR codes. We focus on different cases in order to cover all possible values of parameters d_r , k_c , ρ and σ .

We denote the whole set of blocks in a DSS by \mathcal{B} , where $|\mathcal{B}| = b$. A failed node in block i can be recovered from any $b - \sigma$ blocks. Denoting an instance of such blocks by \mathcal{B}_i^r , we have $|\mathcal{B}_i^r| = b - \sigma$ and $i \notin \mathcal{B}_i^r$. This repair process requires a total number of d nodes and equal number of nodes from each block in \mathcal{B}_i^r , $d_r = \frac{d}{b-\sigma}$, is contacted. Data collector, on the other hand, connects to $b - \rho$ blocks represented by \mathcal{B}^c to reconstruct the stored data. In this process, a total number of k nodes are connected to retrieve the stored data and the same number of nodes from each block in \mathcal{B}^c , $k_c = \frac{k}{b-\rho}$, are connected.

In the following, we analyze the information flow graph for a total number of $k_c(b - \rho)$ failures and repairs, and characterize the min-cut for the corresponding graphs to upper bound the file size that can be stored in DSS.

Our analysis here differs from the classical setup considered in [1], in the sense that the scenarios considered here require analysis of different cases (different min-cuts in information flow graph) depending on the relation between d_r and k_c ($d_r \geq k_c$ vs. $d_r < k_c$). Before detailing the analysis for each case (in Section III-A and III-B), we here point out why this phenomenon occurs. First, consider the case $d_r \geq k_c$, a failed node in block i requires connections from the blocks $j \in \mathcal{B}_i^r$ and that are previously failed and repaired and connected to DC. At this point, block i needs to contact some additional nodes in block j ($d_r - k_c$ number of nodes) that may not be connected to DC. On the other hand, when $d_r < k_c$, then, for the min-cut, any failed node

in block i would contact d_r nodes in block j , which are already connected to DC. Therefore, the storage systems exhibits different min-cuts depending on the values of d_r , k_c , ρ and σ . In the following, we analyze each case separately, obtain the min-cuts as well as corresponding corner points in per-node storage and repair bandwidth trade-off. Section III-A focuses on $d_r \geq k_c$ case, where the number of helper nodes from each block is greater than or equal to the number of nodes connected to DC from each block. Section III-B details the remaining case, i.e., $d_r < k_c$.

A. Case I: $d_r \geq k_c$

1) *Case I.A: $\sigma \leq \rho$:* The first case we focus on is having $\sigma \leq \rho$, which implies that the number of blocks participating to the repair process is greater than or equal to the number of blocks that are connected to DC, i.e., $b - \sigma \geq b - \rho$.

Theorem 13. *If $\alpha = \frac{\mathcal{M}}{k}$ or $\alpha = d\beta$, the upper bound on the file size when $d_r \geq k_c$ and $\sigma \leq \rho$ is given by*

$$\mathcal{M} \leq \sum_{i=1}^{b-\rho} k_c \min \{ \alpha, (d - (i - 1)k_c)\beta \}. \quad (1)$$

Proof. Denote the order of node repairs with corresponding (node) indices in the ordered set \mathcal{O} , where $|\mathcal{O}| = k_c(b - \rho)$. We will show that any order of failures results in the same cut value. We have $\mathcal{C}_{\mathcal{O}} = \sum_{i=1}^{b-\rho} k_c \min \{ \alpha, (d - (i - 1)k_c)\beta \}$. This follows by considering that block i connects to $i - 1$ blocks, each having k_c nodes that are repaired and connected to DC, which makes the cut value of $(d - (i - 1)k_c)\beta$. We denote the stored data of node j in block l as $\mathcal{X}_{l,j}$ and downloaded data to this node as $\mathcal{R}_{l,j}$. The set \mathcal{O} includes an ordered set of node incides (i, j) contacted by DC, where the order indicates the sequence of failed and repaired nodes. (For instance, $\mathcal{O} = \{(1, 1), (1, 2), (2, 3), \dots\}$ refers to a scenario in which third node of second blocks is repaired after second node of first block.) We consider DC connecting to the nodes in

$$\mathcal{O} = \{(1, 1), (1, 2), \dots, (1, k_c), (2, 1), (2, 2), \dots, (2, k_c), \dots, (b - \rho, 1), \dots, (b - \rho, k_c)\},$$

where each block has repairs sequentially. Due to data reconstruction property, it follows that $H(\mathcal{F}|\mathcal{X}_{\mathcal{O}}) = 0$. Accordingly, we have

$$\mathcal{M} = H(\mathcal{F}) = H(\mathcal{F}) - H(\mathcal{F}|\mathcal{X}_{\mathcal{O}}) = I(\mathcal{F}; \mathcal{X}_{\mathcal{O}}) \leq H(\mathcal{X}_{\mathcal{O}}).$$

We consider the following bound on the latter term $H(\mathcal{X}_{\mathcal{O}})$ to obtain $\mathcal{C}_{\mathcal{O}}$, i.e., “cut value” for the repair order given by \mathcal{O} . (Note that the bounds given below correspond to introducing “cuts” in the corresponding information flow graph.) We denote $\mathcal{O}(i) \triangleq \{(i, 1), (i, 2), \dots, (i, k_c)\}$ as the nodes contacted in i -th block. We consider each repaired node contacts the previously repaired nodes in \mathcal{O} . Accordingly, we consider the entropy calculation given by

$$\begin{aligned} H(\mathcal{X}_{\mathcal{O}}) &= \sum_{i=1}^{b-\rho} H(\mathcal{X}_{\mathcal{O}(i)}|\mathcal{X}_{\mathcal{O}(1)}, \dots, \mathcal{X}_{\mathcal{O}(i-1)}) \leq \sum_{i=1}^{b-\rho} \sum_{(i,j) \in \mathcal{O}(i)} H(\mathcal{X}_{(i,j)}|\mathcal{X}_{\mathcal{O}(1)}, \dots, \mathcal{X}_{\mathcal{O}(i-1)}) \quad (2) \\ &\stackrel{(a)}{\leq} \sum_{i=1}^{b-\rho} \sum_{(i,j) \in \mathcal{O}(i)} \min \{ \alpha, (d - (i - 1)k_c)\beta \} \stackrel{(b)}{=} \sum_{i=1}^{b-\rho} k_c \min \{ \alpha, (d - (i - 1)k_c)\beta \} \triangleq \mathcal{C}_{\mathcal{O}} \end{aligned}$$

where (a) follows as $H(\mathcal{X}_{(i,j)}|\mathcal{X}_{\mathcal{O}(1)}, \dots, \mathcal{X}_{\mathcal{O}(i-1)}) \leq H(\mathcal{X}_{(i,j)}) \leq \alpha$ and

$$H(\mathcal{X}_{(i,j)}|\mathcal{X}_{\mathcal{O}(1)}, \dots, \mathcal{X}_{\mathcal{O}(i-1)}) \leq H(\mathcal{R}_{(i,j)}|\mathcal{X}_{\mathcal{O}(1)}, \dots, \mathcal{X}_{\mathcal{O}(i-1)}) \leq (d - (i - 1)k_c)\beta,$$

as $i - 1$ blocks containing previously repaired nodes contributes to $(i - 1)k_c\beta$ symbols in $\mathcal{R}_{(i,j)}$. Therefore, as the bound on $H(X_{(i,j)})$ above holds for any j such that $(i, j) \in \mathcal{O}_{(i)}$, we have (b), from which we obtain the following bound on file size \mathcal{M}

$$\mathcal{M} \leq \sum_{i=1}^{b-\rho} k_c \min \{ \alpha, (d - (i - 1)k_c)\beta \}.$$

Interchanging the failure order in \mathcal{O} at indices k_c and $k_c + 1$, we obtain another order \mathcal{O}^* . Using the same bounding technique above, the resulting cut value can be calculated as $\mathcal{C}_{\mathcal{O}^*} = (k_c - 1) \min \{ \alpha, d\beta \} + \min \{ \alpha, (d - (k_c - 1))\beta \} + \min \{ \alpha, (d - 1)\beta \} + (k_c - 1) \min \{ \alpha, (d - k_c)\beta \} + \sum_{i=3}^{b-\rho} k_c \min \{ \alpha, (d - (i - 1)k_c)\beta \}$. Note that the first term here corresponds to the first set of $k_c - 1$ nodes in the first blocks, second term corresponds to the first node repaired in the second block, third term corresponds to the remaining node in the first block, fourth terms corresponds to remaining $k_c - 1$ nodes in the second block, and the last term corresponds to the remaining blocks. We observe that $\mathcal{C}_{\mathcal{O}} = \mathcal{C}_{\mathcal{O}^*}$, i.e., the total cut values induced by \mathcal{O} and \mathcal{O}^* are the same. Consider $\mathcal{C}_{\mathcal{O}^*} - \mathcal{C}_{\mathcal{O}}$, which evaluates to

$$\mathcal{C}_{\mathcal{O}^*} - \mathcal{C}_{\mathcal{O}} = \min \{ \alpha, (d - 1)\beta \} - \min \{ \alpha, d\beta \} + \min \{ \alpha, (d - (k_c - 1))\beta \} - \min \{ \alpha, (d - k_c)\beta \}.$$

We observe that $\mathcal{C}_{\mathcal{O}^*} - \mathcal{C}_{\mathcal{O}} = 0$ at both $\alpha = \frac{\mathcal{M}}{k}$ and $\alpha = d\beta$ operating points. (For $\alpha = \frac{\mathcal{M}}{k}$, all $\min \{ \}$ terms result in α and cancel each other. For the $\alpha = d\beta$ point, $\min \{ \}$ terms remove α (as $\alpha = d\beta$) and remaining terms cancel each other.)

The equivalence analysis above can be extended to any pair of orders. In the following, we show this through the following lemmas. We first observe that any order can be obtained from another by swapping adjacent failures.

Lemma 14. *Any given failure order \mathcal{O}^* can be obtained by permuting the elements in the order \mathcal{O} , and the underlying permutation $\pi_{\mathcal{O} \rightarrow \mathcal{O}^*}$ operation can be decomposed into stages of swapping of adjacent elements.*

We provide an example here, the lemma above generalizes this argument to any pair of orders. Consider $\mathcal{O} = (1, 2, 3)$ and $\mathcal{O}^* = (3, 2, 1)$, the permutation $\pi_{\mathcal{O} \rightarrow \mathcal{O}^*}$ is given by position mappings $\{1 \rightarrow 3, 2 \rightarrow 2, 3 \rightarrow 1\}$. This can be obtained by composition of three permutations $\{1 \rightarrow 1, 2 \rightarrow 3, 3 \rightarrow 2\}$, for swapping 2 and 3, $\{1 \rightarrow 2, 2 \rightarrow 1, 3 \rightarrow 3\}$, for swapping 1 and 3, and $\{1 \rightarrow 1, 2 \rightarrow 3, 3 \rightarrow 2\}$, for swapping 1 and 2.

Utilizing Lemma 14, it remains to show that the cut value remains the same if we interchange adjacent failures in any failure order \mathcal{O} . We show that this holds in the following lemma.

Lemma 15. *For any order \mathcal{O} swapping any adjacent failures does not result in min-cut value $\mathcal{C}_{\mathcal{O}}$ to change.*

To show this, consider two orders \mathcal{O}^1 and \mathcal{O}^2 , where \mathcal{O}^2 is obtained by swapping order of failures at locations j and $j+1$, $\mathcal{O}^2 = \pi(\mathcal{O}^1)$. Then, we can say that the cut values up to $j-1$ and the cut values after $j+1$ are individually the same for both \mathcal{O}^1 and \mathcal{O}^2 . Furthermore, there are two possible cases for swapped failures j and $j+1$; i) either both are from the same block, ii) they are from different blocks. For the former case, the swapping does not affect the cut values for failures j and $j+1$. In the latter, first note that when $\alpha = \frac{\mathcal{M}}{k}$, there is no change in cut value, $\mathcal{C}_{\mathcal{O}^1} = \mathcal{C}_{\mathcal{O}^2}$, hence we'll only focus on $\alpha = d\beta$ case. Assume for the \mathcal{O}^1 we have $(d-i)\beta$ for i^{th} and $(d-i^*)\beta$ for $(i+1)^{\text{th}}$ failures. Then, \mathcal{O}^2 should have $(d-(i^*-1))\beta$ and $(d-(i+1))\beta$ respectively. Note that the sums are still the same, $(d-i)\beta + (d-i^*)\beta = (d-(i^*-1))\beta + (d-(i+1))\beta$, hence we can conclude that swapping any two adjacent failures does not change the min-cut value, $\mathcal{C}_{\mathcal{O}^1} = \mathcal{C}_{\mathcal{O}^2}$.

Combining two lemmas, we conclude that every order of failures has the same cut value as \mathcal{O} , which is $\mathcal{C}_{\mathcal{O}}$. \square

The key distinction to be made in our scenario is that all of the nodes that are failed and repaired are utilized in the repair process of the subsequent failures, which is why the order of failures does not matter. Note that for the special case of $k_c = 1$, the above bound reduces to the classical bound given in [1], albeit only for $\alpha = \frac{\mathcal{M}}{k}$ or $\alpha = d\beta$ case. We obtain the following corner points in the trade-off region.

Corollary 16. *For $d_r \geq k_c$ and $\sigma \leq \rho$, corresponding BFR-MSR and BFR-MBR points can be found as follows.*

$$(\alpha_{\text{BFR-MSR}}, \gamma_{\text{BFR-MSR}}) = \left(\frac{\mathcal{M}}{k}, \frac{\mathcal{M}d}{kd - \frac{k^2(b-\rho-1)}{b-\rho}} \right), \quad (3)$$

$$(\alpha_{\text{BFR-MBR}}, \gamma_{\text{BFR-MBR}}) = \left(\frac{\mathcal{M}d}{kd - \frac{k^2(b-\rho-1)}{2(b-\rho)}}, \frac{\mathcal{M}d}{kd - \frac{k^2(b-\rho-1)}{2(b-\rho)}} \right). \quad (4)$$

Proof. For BFR-MSR point, we set $\alpha = \frac{\mathcal{M}}{k}$ in (1) and obtain the requirements $[d - (b - \rho - 1)k_c\beta \geq \alpha]$ from which we obtain that minimum γ occurs at $\gamma_{\text{BFR-MSR}} = \frac{\alpha}{d - (b - \rho - 1)k_c}d$ as β is lower bounded by $\frac{\alpha}{d - (b - \rho - 1)k_c}$. BFR-MBR point, on the other hand, follows from the bound $\mathcal{M} \leq \sum_{i=1}^{b-\rho} k_c[d - (i-1)k_c]\beta$ as $\alpha = d\beta$ at the minimum bandwidth. More specifically, minimum β is given by $\beta = \frac{\mathcal{M}}{\sum_{i=1}^{b-\rho} k_c[d - (i-1)k_c]}$ and $\gamma_{\text{BFR-MBR}} = \alpha_{\text{BFR-MSR}} = d\beta$. \square

Remark 17. *When $b = \rho + 1$, we observe that $\alpha_{\text{BFR-MSR}} = \gamma_{\text{BFR-MSR}} = \alpha_{\text{BFR-MBR}} = \gamma_{\text{BFR-MBR}} = \frac{\mathcal{M}}{k}$.*

2) *Case I.B: $\sigma > \rho$:* For the case of having $\sigma > \rho$, we have $|\mathcal{B}^c| = b - \rho > |\mathcal{B}^r| = b - \sigma$. Noting that the helper nodes are chosen from the ones that are already connected to DC, we consider that $\mathcal{B}^c \supset \mathcal{B}^r$. Here, the min-cut analysis similar to the one given in the proof of Lemma 18 is same as having a system with $\bar{b} = b - \rho$, $\bar{\rho} = 0$, $\bar{\sigma} = \sigma - \rho$ since the analysis for the file size bound only utilizes $b - \rho$ blocks and repairs occur by connecting to a subset of these $b - \rho$ blocks.

That is, the remaining ρ blocks do not contribute to the cut value. Therefore, we conclude that $\mathcal{C}(b, \rho, \sigma) = \mathcal{C}(b - \rho, 0, \sigma - \rho)$ when $\sigma > \rho$.

Lemma 18. *An upper bound on the file size when $d_r \geq k_c$ and $\sigma > \rho$ is given by*

$$\mathcal{M} \leq \sum_{i=1}^{b-\sigma} k_c \min \{\alpha, \beta(d - (i-1)k_c)\} + \sum_{i=b-\sigma+1}^{b-\rho} k_c \min \{\alpha, \beta(d - (b-\sigma)k_c)\}. \quad (5)$$

Proof. Let \mathcal{O} be an order such that first k_c failures occur in the first block, the next k_c failures occur in the second block and so on. Denote by $\mathcal{C}_{\mathcal{O}}$ the total cut value induced by \mathcal{O} . (The analysis detailed in the proof of Theorem 13 is followed here.) Consider the node indexed by $i = k_c(b - \sigma)$ so that order \mathcal{O} can be split into two parts as \mathcal{O}^{i-} and \mathcal{O}^{i+} , where \mathcal{O}^{i-} represents the failures up to (and including) the node i , and \mathcal{O}^{i+} represents the remaining set of failures.

Since any node failure contacts $b - \sigma$ blocks and noting that \mathcal{O}^{i-} includes exactly $b - \sigma$ blocks, any node failure in \mathcal{O}^{i+} would contribute to the cut value as $\min \{\alpha, (d - (b - \sigma)k_c)\beta\}$. On the other hand, $\mathcal{C}_{\mathcal{O}^{i-}}$ follows from Theorem 13. Combining $\mathcal{C}_{\mathcal{O}^{i-}}$ and $\mathcal{C}_{\mathcal{O}^{i+}}$, we get (5). \square

Remark 19. *We conjecture that the order given in the proof above corresponds to the order producing the min-cut. We verified this with numerical analysis for systems having small n values. Although a general proof of this conjecture is not established yet ², we were able to construct codes achieving the stated bound. Therefore we conjecture the following MSR/MBR points for this case.*

Utilizing the bound given in Lemma 18, we obtain following result (proof is similar to Corollary 16 and omitted for brevity).

Conjecture 20. *For $d_r \geq k_c$ and $\sigma > \rho$, corresponding BFR-MSR and BFR-MBR points can be found as follows.*

$$(\alpha_{BFR-MSR}, \gamma_{BFR-MSR}) = \left(\frac{\mathcal{M}}{k}, \frac{\mathcal{M}d}{kd - \frac{k^2(b-\sigma)}{b-\rho}} \right), \quad (6)$$

$$(\alpha_{BFR-MBR}, \gamma_{BFR-MBR}) = \left(\frac{\mathcal{M}d}{kd - \frac{k^2(b-\sigma)(b+\sigma-2\rho-1)}{2(b-\rho)^2}}, \frac{\mathcal{M}d}{kd - \frac{k^2(b-\sigma)(b+\sigma-2\rho-1)}{2(b-\rho)^2}} \right). \quad (7)$$

If code constructions achieve the points above ((6) and (7)), then this will result in an optimal BFR-MSR/BFR-MBR. Later, we propose code constructions achieving these points, see Section IV-C.

²The main difficulty for this case is having $\sigma > \rho$, which makes some failed nodes not being utilized in the repair process. That is why the proposed order is constructed such a way that we try to maximize the use of failed nodes in the repair process of subsequent nodes.

B. Case II: $d_r < k_c$

We first note that $d \geq k$. (This follows similarly to the analysis for regenerating codes, as otherwise one can regenerate each node by contacting $d < k$ nodes, and obtain the stored data with less than k nodes.) Assume $\rho \leq \sigma$, then $b - \sigma \leq b - \rho$. This, together with $d_r < k_c$ implies that repair operation contacts to less number of blocks and less number of nodes per block as compared to data access. If this is the case, via a repair operation, all the remaining nodes can be regenerated and data access can be completed using repair operation. Therefore, the valid scenario is $\rho > \sigma$. (This is similar to the reasoning of $d \geq k$ assumption in regenerating codes.)

Theorem 21. *The optimal file size when $d_r < k_c$ and $\rho > \sigma$ is given by*

$$\mathcal{M} \leq \sum_{i=1}^{b-\rho} d_r \min \{ \alpha, (d - (i-1)d_r)\beta \} + \sum_{i=1}^{b-\rho} (k_c - d_r) \min \{ \alpha, (d - (b - \rho - 1)d_r)\beta \}. \quad (8)$$

Proof. Let \mathcal{O} be an order and let index $i = d_r(b - \rho)$ so that order \mathcal{O} can be split into two parts as \mathcal{O}^{i-} and \mathcal{O}^{i+} where \mathcal{O}^{i-} represents the failures up to (and including) index i and \mathcal{O}^{i+} represents the remaining set of failures. For index i , $\mathcal{C}_{\mathcal{O}^{i-}}$ takes its minimum possible value if \mathcal{O}^{i-} contains exactly d_r failures from each of $b - \rho$ blocks. We show this by a contradiction. Assume that \mathcal{O}^{i-} contains d_r failures from each of $b - \rho$ blocks but corresponding $\mathcal{C}_{\mathcal{O}^{i-}}$ is not the minimum. We have already shown that the failure order among the nodes in \mathcal{O}^{i-} does not matter as long as the list contains d_r failures from $b - \rho$ blocks. This follows from $d_r = k_c$ case analyzed in Theorem 13. Assume an order spanning $b - \rho + t$ blocks for $t > 0$. This ordering will include nodes that are not connected to DC and can be omitted. This means that an order that minimizes the cut value needs to contain a block that has at least $d_r + 1$ failures. Denote such an ordering by \mathcal{O}^{i--} , assume without loss of generality that this block j has $t \geq 1$ additional failures. In such a case, we observe that the cut value for failed nodes in other blocks would not be affected by this change since each such node is already connected to d_r nodes of block j . Furthermore, by removing a failed node of some other block (other than j) from \mathcal{O}^{i-} , the cut values corresponding to other nodes in the list would only increase since the failures can benefit at most $d_r - 1$ nodes of that block as opposed to d_r in \mathcal{O}^{i-} . Hence, in order to minimize $\mathcal{C}_{\mathcal{O}^{i-}}$, \mathcal{O}^{i-} needs to include exactly d_r failures from each of $b - \rho$ blocks. Also, it can be observed that, when \mathcal{O}^{i-} is constructed as above, then \mathcal{O}^{i+} also takes its minimum possible value, since any failure in \mathcal{O}^{i+} utilizes maximum possible number of repaired nodes (d_r nodes from each of the $b - \rho - 1$ blocks) that are connected to DC hence it only needs to contact $\rho - \sigma + 1$ blocks that are not connected to DC. Therefore, both $\mathcal{C}_{\mathcal{O}^{i-}}$ and $\mathcal{C}_{\mathcal{O}^{i+}}$ are minimized individually with \mathcal{O}^{i-} . Therefore, for a fixed threshold $i = d_r(b - \rho)$, we obtain the min-cut. \square

Corollary 22. *For $d_r < k_c$, corresponding BFR-MSR and BFR-MBR points can be found as follows.*

$$(\alpha_{BFR-MSR}, \gamma_{BFR-MSR}) = \left(\frac{\mathcal{M}}{k}, \frac{\mathcal{M}d}{\frac{kd(\rho-\sigma+1)}{b-\sigma}} \right), \quad (9)$$

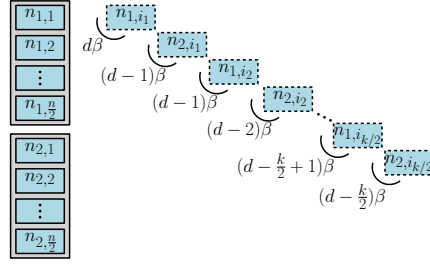


Fig. 3: Repair process for $b = 2$ (two blocks) case.

$$(\alpha_{BFR-MBR}, \gamma_{BFR-MBR}) = \left(\frac{\mathcal{M}d}{\frac{kd(\rho-\sigma+1)}{b-\sigma} + \frac{d^2(b-\rho)(b-\rho-1)}{2(b-\sigma)^2}}, \frac{\mathcal{M}d}{\frac{kd(\rho-\sigma+1)}{b-\sigma} + \frac{d^2(b-\rho)(b-\rho-1)}{2(b-\sigma)^2}} \right). \quad (10)$$

C. BFR-MSR and BFR-MBR Points for Special Cases

The general case is analyzed in the previous section, and we here focus on special cases for BFR-MSR and BFR-MBR points. The first case analyzed below has the property that corresponding BFR-MSR codes achieve both per-node storage point of MSR codes and repair bandwidth of MBR codes simultaneously when $2d \gg k$.

1) *Special Case for I.B:* $\rho = 0$, $\sigma = 1$, $d_r \geq k_c$ and $b = 2$: Consider the 2-block case ($b = 2$) as in Fig. 3, and assume $2 \mid k$. The file size \mathcal{M} can be upper bounded with the repair procedure shown in Fig. 3, which displays one of the “minimum-cut” scenarios, wherein any two consecutive node failures belong to different blocks. Assuming $d \geq \frac{k}{2}$, we obtain

$$\mathcal{M} \leq \sum_{i=0}^{\frac{k}{2}-1} \min\{\alpha, (d-i)\beta\} + \sum_{i=1}^{\frac{k}{2}} \min\{\alpha, (d-i)\beta\}. \quad (11)$$

Achieving this upper bound (11) with equality would yield maximum possible file size. One particular repair instance is shown in Fig. 3, and we note that the order of node repairs does not matter as the sum of the corresponding cut values would be the same with different order of failures as long as we consider connection from data collector to $\frac{k}{2}$ repaired nodes from each block.

For BFR-MSR point, $\alpha = \alpha_{BFR-MSR} = \frac{\mathcal{M}}{k}$. In the bound (11), we then have $\alpha_{BFR-MSR} \leq (d - \frac{k}{2})\beta_{BFR-MSR}$. Achieving equality would give the minimum repair bandwidth for the MSR case. Hence, BFR-MSR point is given by

$$(\alpha_{BFR-MSR}, \gamma_{BFR-MSR}) = \left(\frac{\mathcal{M}}{k}, \frac{2\mathcal{M}d}{2kd - k^2} \right). \quad (12)$$

Note that, this coincides with that of (6) where we set $b = 2$, $\rho = 0$ and $\sigma = 1$ therein.

BFR-MBR codes, on the other hand, have the property that $d\beta = \alpha$ with minimum possible $d\beta$ while achieving the equality in (11). Inserting $d\beta = \alpha$ in (11), we obtain that

$$(\alpha_{BFR-MBR}, \gamma_{BFR-MBR}) = \left(\frac{4\mathcal{M}d}{4dk - k^2}, \frac{4\mathcal{M}d}{4dk - k^2} \right). \quad (13)$$

This coincides with that of (7) where we set $b = 2$, $\rho = 0$ and $\sigma = 1$ therein.

We now consider the case where $2 \nmid k$ (as compared to previous section where we assumed $k_c = \frac{k}{b-\rho}$), and characterize trade-off points for all possible system parameters in this special case. First consider the special case of $k = 3$ and two different order of failures, one with first failure in first block, second failure in second block, third failure in first block and the other one with first and second failures from first block, third failure from second block. Accordingly, observe that the cuts as $\min\{\alpha, d\beta\} + 2\min\{\alpha, (d-1)\beta\}$ and $2\min\{\alpha, d\beta\} + \min\{\alpha, (d-2)\beta\}$ respectively. For MSR case, first sum would require $\alpha = (d-1)\beta$, whereas second sum requires $\alpha = (d-2)\beta$, resulting in higher repair bandwidth. Henceforth, one needs to be careful even though cut values are same for both orders of failures in both MSR (3α) and MBR ($(3d-2)\beta$) cases.

$$\mathcal{M} \leq \sum_{i=1}^{\lfloor \frac{k}{2} \rfloor + 1} \min\{\alpha, d\beta\} + \sum_{i=1}^{\lfloor \frac{k}{2} \rfloor} \min\{\alpha, (d - \lfloor \frac{k}{2} \rfloor - 1)\beta\} \quad (14)$$

The corresponding trade-off point are summarized below by following analysis similar to the one above.

$$(\alpha_{\text{BFR-MSR}}, \gamma_{\text{BFR-MSR}}) = \begin{cases} (\frac{M}{k}, \frac{2Md}{2kd-k^2-k}), & \text{if } k \text{ is odd} \\ (\frac{M}{k}, \frac{2Md}{2kd-k^2}), & \text{o.w.} \end{cases} \quad (15)$$

$$(\alpha_{\text{BFR-MBR}}, \gamma_{\text{BFR-MBR}}) = \begin{cases} (\frac{4Md}{4dk-k^2+1}, \frac{4Md}{4dk-k^2+1}), & \text{if } k \text{ is odd} \\ (\frac{4Md}{4dk-k^2}, \frac{4Md}{4dk-k^2}), & \text{o.w.} \end{cases} \quad (16)$$

Here, we compare $\gamma_{\text{BFR-MSR}}$ and γ_{MBR} . We have $\min\{\gamma_{\text{BFR-MSR}}^{\text{k-odd}}, \gamma_{\text{BFR-MSR}}^{\text{k-even}}\} \geq \gamma_{\text{MBR}} = \frac{2Md}{k(2d-k+1)}$, and, if we have $2d - k \gg 1$, then $\gamma_{\text{BFR-MSR}}^{\text{k-odd}} \approx \gamma_{\text{BFR-MSR}}^{\text{k-even}} \approx \gamma_{\text{MBR}}$. This implies that BFR-MSR codes with $b = 2$ achieves repair bandwidth of MBR and per-node storage of MSR codes simultaneously for systems with $d \gg 1$. On Fig. 4a and 4b, we depict the ratio of $\gamma_{\text{BFR-MSR}}^{\text{k-odd}}$ and $\gamma_{\text{BFR-MSR}}^{\text{k-even}}$ to γ_{MBR} respectively, where we keep k constant and vary d as $2k \geq d \geq k$. Also, only even d values are shown in both figures. It can be observed that ratio gets closer to 1 as we increase k . Next, we provide the generalization of critical points to $b \geq 2$ case in the following.

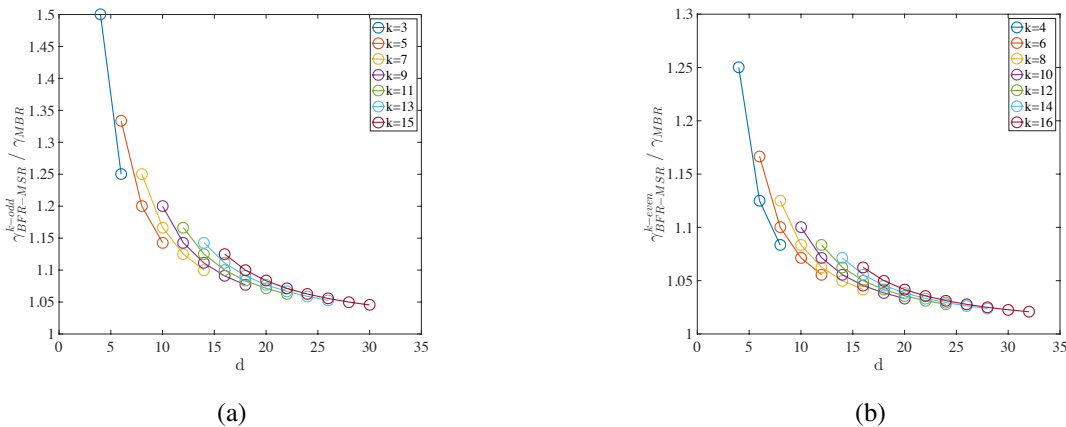


Fig. 4: (a) Ratio $\frac{\gamma_{\text{BFR-MSR}}^{\text{k-odd}}}{\gamma_{\text{MBR}}}$ vs. d . (b) Ratio $\frac{\gamma_{\text{BFR-MSR}}^{\text{k-even}}}{\gamma_{\text{MBR}}}$ vs. d .

Remark 23. The bound in (11) holds for the intermediate points as well for the special case discussed in this section ($b = 2, \rho = 0, \sigma = 1$) (as compared to previous section where the general case is analyzed). See Fig. 2(c) and [42].

2) *Special Case for Case I.B:* $\rho = 0, \sigma = 1, d_r \geq k_c$ and $b \geq 2$: From Corollary 20, we obtain the corresponding MSR and MBR points in this special case.

Corollary 24. For $\rho = 0, \sigma = 1, d_r \geq k_c$ and $b \geq 2$ BFR-MSR and BFR-MBR points are as follows.

$$(\alpha_{\text{BFR-MSR}}, \gamma_{\text{BFR-MSR}}) = \left(\frac{\mathcal{M}}{k}, \frac{\mathcal{M}d}{kd - \frac{k^2(b-1)}{b}} \right) \quad (17)$$

$$(\alpha_{\text{BFR-MBR}}, \gamma_{\text{BFR-MBR}}) = \left(\frac{\mathcal{M}d}{kd - \frac{k^2(b-1)}{2b}}, \frac{\mathcal{M}d}{kd - \frac{k^2(b-1)}{2b}} \right) \quad (18)$$

We observe that $\gamma_{\text{BFR-MSR}} \leq \gamma_{\text{MSR}} = \frac{\mathcal{M}d}{k(d-k+1)}$ for $b \leq k$, which is the case here as we assume $b \mid k$. Also, we have $\frac{\gamma_{\text{BFR-MSR}}}{\gamma_{\text{MBR}}} = \frac{d - \frac{k-1}{2}}{d - k\frac{b-1}{b}} \geq 1$ when $b \geq \frac{2k}{k+1}$ which is always true. Hence, $\gamma_{\text{BFR-MSR}}$ is between γ_{MSR} and γ_{MBR} , see Fig. 2(c).

IV. BFR-MSR AND BFR-MBR CODE CONSTRUCTIONS

A. Transpose code: $b=2$

Construction I (Transpose code): Consider $\alpha = d = \frac{n}{2}$, and placement of $\frac{n\alpha}{2}$ symbols denoted by $\{x_{i,j} : i, j \in \{1, 2, \dots, \frac{n}{2} = \alpha\}\}$ for $b = 2$ blocks according to the following rule: Node i in (in block $b = 1$) stores symbols $\{x_{i,j} : j \in \{1, 2, \dots, \alpha\}\}$, whereas node $i + \frac{n}{2}$ (in block $b = 2$) stores symbols $\{x_{j,i} : j \in \{1, 2, \dots, \alpha\}\}$ for $i = 1, 2, \dots, \frac{n}{2}$. Note that, when the stored symbols in nodes of block 1 is represented as a matrix, the symbols in block 2 corresponds to transpose of that matrix. (We therefore refer to this code as transpose code.)

Due to this transpose property, the repair of a failed node i in the first block can be performed by connecting to all the nodes in the second block and downloading only 1 symbol from each node. That is, we have $d\beta = \alpha$. Consider now that the file size $\mathcal{M} = kd - (\frac{k}{2})^2$, and an $[N = \alpha^2, K = \mathcal{M}]$ MDS code is used to encode file \mathbf{f} into symbols denoted with $x_{i,j}, i, j = 1, \dots, \alpha$. Here, BFR data collection property for reconstructing the file is satisfied, as connecting to any $k_c = \frac{k}{2}$ nodes from each block assures at least K distinct symbols. This can be shown as follows: Consider $\alpha \times \alpha$ matrix X , where i -th row, j -th column has the element $x_{i,j}$. Rows of X correspond to nodes of block 1, and columns of X correspond to nodes of block 2. Any $\frac{k}{2}$ rows (or any $\frac{k}{2}$ columns) provide total of $\frac{k\alpha}{2}$ symbols. And $\frac{k}{2}$ rows and $\frac{k}{2}$ columns intersect at $(\frac{k}{2})$ symbols. Therefore, total number of symbols from any $\frac{k}{2}$ rows and $\frac{k}{2}$ columns is \mathcal{M} . Note that the remaining system parameters are $d_r = \frac{n}{2} \geq k_c, \rho = 0$ and $\sigma = 1$. Henceforth, this code is a BFR-MBR code as the operating point in (18), is achieved with $\mathcal{M} = kd - (\frac{k}{2})^2$ and $d\beta = \alpha$ for $\beta = 1$ (scalar code).

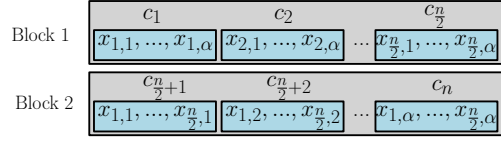


Fig. 5: Transpose code is a two-block BFR-MBR code.

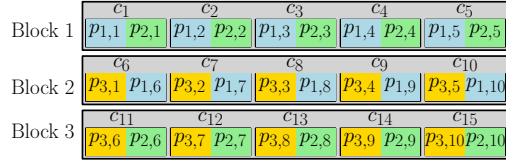
A similar code to this construction is Twin codes introduced in [24], where the nodes are split into two types and a failed node of a given type is regenerated by connecting to nodes only in the other type. During construction of Twin codes, the message is first transposed and two different codes are applied to both original message and its transposed version separately to obtain code symbols. On the other hand, we apply one code to the message and transpose resulting symbols during placement. Also, Twin codes, as opposed to our model, do not have balanced node connection for data collection. In particular, DC connects to only (a subset of k nodes from) a single type and repairs are conducted from k nodes. On the other hand, BFR codes, for $b = 2$ case, connects to $\frac{k}{2}$ nodes from each block and repairs are from any d nodes in other block.

This construction, however, is limited to $b = 2$ and in the following section we propose utilization of block designs to construct BFR codes for $b > 2$.

B. Projective plane based placement of regenerating codewords ($\rho = 0, \sigma = 1$)

$$\begin{bmatrix} 1 & 2 \\ 3 & 1 \\ 3 & 2 \end{bmatrix}$$

(a)



(b)

Fig. 6: (a) Matrix representation of block design. (b) Three-block BFR-RC.

Consider that the file \mathbf{f} of size \mathcal{M} is partitioned into 3 parts \mathbf{f}_1 , \mathbf{f}_2 and \mathbf{f}_3 each of size $\tilde{\mathcal{M}} = \frac{\mathcal{M}}{3}$. Each partition \mathbf{f}_i is encoded with an $[\tilde{n} = 10, \tilde{k} = 4, \tilde{d} = 5, \tilde{\alpha}, \tilde{\beta}]$ regenerating code $\tilde{\mathcal{C}}$, where the resulting partition codewords are represented with $\mathcal{P}_1 = \{p_{1,1:\tilde{n}}\}$ for \mathbf{f}_1 , $\mathcal{P}_2 = \{p_{2,1:\tilde{n}}\}$ for \mathbf{f}_2 , and $\mathcal{P}_3 = \{p_{3,1:\tilde{n}}\}$ for \mathbf{f}_3 . These symbols are grouped in a specific way and placed into nodes within blocks as represented in Fig. 6, where each node contains two symbols each coming from two different partitions. We set the BFR code parameters as $[\mathcal{M} = 3\tilde{\mathcal{M}}, k = \frac{3}{2}\tilde{k}, d = 2\tilde{d}, \alpha = 2\tilde{\alpha}, \beta = \tilde{\beta}]$.

Assume the first block (denoted as Block 1) is unavailable and its first node, which contains codeword c_1 , has to be reconstructed. Due to underlying regenerating code, contacting 5 nodes of Block 2 and accessing to $p_{1,6:10}$ regenerates $p_{1,1}$. Similarly, $p_{2,1}$ can be reconstructed from Block 3. Any node reconstruction can be handled similarly, by connecting to remaining 2 blocks and repairing each symbol of the failed node by corresponding \tilde{d} nodes in each block. As we

have $k = 6$, DC, by connecting to 2 nodes from each block, obtains a total of 12 symbols, which²⁰ consist of 4 different symbols from each of \mathcal{P}_1 , \mathcal{P}_2 and \mathcal{P}_3 . As the embedded regenerating code has $\tilde{k} = 4$, all 3 partitions ($\mathbf{f}_1, \mathbf{f}_2$ and \mathbf{f}_3) can be recovered, from which \mathbf{f} can be reconstructed.

In the following construction, we generalize the BFR-RC construction above utilizing projective planes for the case of having $\rho = 0$, $\sigma = 1$. As defined in Section III, this necessarily requires $d_r > k_c$.) We first introduce projective planes in the following and then detail the code construction.

Definition 25 (Balanced incomplete block design [43]). *A (v, κ, λ) -BIBD has v points distributed into blocks of size κ such that any distinct pair of points are contained in λ blocks.*

Corollary 26. *For a (v, κ, λ) -BIBD,*

- *Every point occurs in $r = \frac{\lambda(v-1)}{\kappa-1}$ blocks.*
- *The design has exactly $b = \frac{vr}{\kappa} = \frac{\lambda(v^2-v)}{\kappa^2-\kappa}$ blocks.*

In the achievable schemes of this work, we utilize a special class of block designs that are called projective planes [43].

Definition 27. *A $(v = p^2 + p + 1, \kappa = p + 1, \lambda = 1)$ -BIBD with $p \geq 2$ is called a projective plane of order p .*

Projective planes have the property that every pair of blocks intersect at a unique point (as $\lambda = 1$). In addition, due to Corollary 26, in projective planes, every point occurs in $r = p + 1$ blocks, and there are $b = v = p^2 + p + 1$ blocks.

Construction II (Projective plane based placement of regenerating codes): For any $(n, b, \mathcal{M}, k, \rho = 0, \alpha, d, \sigma = 1, \beta)$ code satisfying $b = p^2 + p + 1$, $k = \frac{b}{p+1}\tilde{k}$, $d = (p+1)\tilde{d}$, $\alpha = (p+1)\tilde{\alpha}$, $\beta = \tilde{\beta}$, $p \mid \tilde{d}$, $p+1 \mid \tilde{n}$, $p+1 \mid \tilde{k}$ where p is the order of the underlying projective plane, and $[\tilde{n}, \tilde{k}, \tilde{d}, \tilde{\alpha}, \tilde{\beta}]$ represents the underlying regenerating code parameters, consider a file \mathbf{f} of size \mathcal{M} .

- First divide \mathcal{M} into $v = p^2 + p + 1$ parts, $\mathbf{f}_1, \mathbf{f}_2, \dots, \mathbf{f}_v$.
- Each part, of size $\tilde{\mathcal{M}} = \frac{\mathcal{M}}{v}$, is then encoded using $[\tilde{n}, \tilde{k}, \tilde{d}, \tilde{\alpha}, \tilde{\beta}]$ regenerating code $\tilde{\mathcal{C}}$. We represent the resulting partition codewords with $\mathcal{P}_i = p_{i,1:\tilde{n}}$ for $i = 1, \dots, v$. We then consider index of each partition as a point in a $(v = p^2 + p + 1, \kappa = p + 1, \lambda = 1)$ projective plane. (Indices of symbol sets $\mathcal{P}_{\mathcal{J}}$ and points \mathcal{J} of the projective plane are used interchangeably in the following.)
- We perform the placement of each symbol to the system using this projective plane mapping. (The setup in Fig. 6(b) can be considered as a toy model. Although the combinatorial design with blocks given by $\{p_1, p_2\}, \{p_3, p_1\}, \{p_3, p_2\}$ has projective plane properties with $p = 1$, it is not considered as an instance of a projective plane [43].) In this placement, total of \tilde{n} symbols from each partition \mathcal{P}_i are distributed to r blocks evenly such that each block contains $\frac{\tilde{n}}{r}$ nodes where each node stores $\alpha = \kappa\tilde{\alpha}$ symbols.

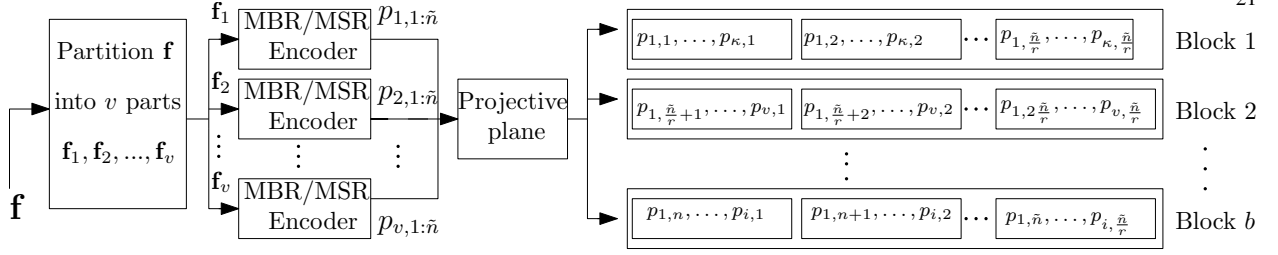


Fig. 7: Illustrating the construction of BFR codes using projective plane based placement of regenerating codes. ($n' = \tilde{n} - \frac{\tilde{n}}{r} + 1$.)

Note that blocks of projective plane give the indices of partitions \mathcal{P}_i stored in the nodes of the corresponding block in DSS. That is, all nodes in a block stores symbols from unique subset of $\mathcal{P} = \{\mathcal{P}_1, \dots, \mathcal{P}_v\}$ of size κ . (For instance, in Fig. 6(b), the first block of the block design has part $\{p_1, p_2\}$, and accordingly symbols from partitions \mathcal{P}_1 and \mathcal{P}_2 are placed into node of Block 1.) Here, as each point in the block design is repeated in r blocks, the partition codewords span r blocks. Overall, the system can store a file of size $\mathcal{M} = v\tilde{\mathcal{M}}$ with $b = v$ blocks. (Note that, $r = \kappa = p + 1$ and $b = v = p^2 + p + 1$ for projective planes. See Definition 27.) We have the parameters as

$$\mathcal{M} = v\tilde{\mathcal{M}}, k = \frac{b}{r}\tilde{k}, d = \kappa\tilde{d}, \alpha = \kappa\tilde{\alpha}, \beta = \tilde{\beta} \quad (19)$$

where we choose the parameters to satisfy $r - 1 \mid \tilde{d}$, $r \mid \tilde{n}$ (for splitting partition codewords evenly to r blocks) and $r \mid \tilde{k}$ (for data collection as detailed below). We have $d_r = \frac{\tilde{d}}{r-1} = \frac{d}{b-1} > k_c = \frac{\tilde{k}}{r} = \frac{k}{b}$ as $d \geq k$ and hence the required condition $d_r > k_c$ is satisfied

Node Repair: Consider that one of the nodes in a block is to be repaired. Note that the failed node contains κ symbols, each coming from a distinct partition. Using the property of projective planes that any 2 blocks has only 1 point in common, any remaining block can help for the regeneration of 1 symbol of the failed node. Furthermore, as any point in the block design has a repetition degree of r , one can connect to $r - 1$ blocks, $d_r = \frac{\tilde{d}}{r-1}$ nodes per block, to repair one symbol of a failed node. Combining these two observations; we observe that node regeneration can be performed by connecting to $(r - 1)\kappa$ blocks. In particular, substituting $r = \kappa = p + 1$, we see that connecting to $p^2 + p = b - 1$ blocks allows for reconstructing of any node of a failed block.

Data Collection: DC connects to $k_c = \frac{\tilde{k}}{r}$ nodes per block from all $b_c = b$ blocks, i.e., a total of $k = \frac{b}{r}\tilde{k} = \frac{v}{r}\tilde{k}$ nodes each having encoded symbols of $\kappa = r$ partitions. These total of $v\tilde{k}$ symbols include \tilde{k} symbols from each partition, from which all partitions can be decoded, and hence the file \mathbf{f} , can be reconstructed.

Remark 28. This construction is related to layered codes studied in [26]. In that work, layering helps to construct codes with exact repair properties. In Construction II, on the other hand, multiple nodes in the system can have the same type (representing the same subset of partitions), and this enables to achieve different operating points for the block failure model.

1) *BFR-MSR*: To construct a BFR-MSR code, we set each sub-code \tilde{C} in Construction II as an MSR code, which has $\tilde{\alpha} = \frac{\tilde{\mathcal{M}}}{\tilde{k}}$ and $\tilde{d}\tilde{\beta} = \frac{\tilde{\mathcal{M}}\tilde{d}}{\tilde{k}(\tilde{d}-\tilde{k}+1)}$. This, together with (19), results in the following parameters of our BFR-MSR construction

$$\alpha = \tilde{\alpha}\kappa = \frac{\mathcal{M}}{k}, d\beta = \kappa\tilde{d}\tilde{\beta} = \frac{\mathcal{M}d}{k(d - \frac{k(p+1)^2}{p^2+p+1} + p + 1)}. \quad (20)$$

We remark that if we utilize ZigZag codes [2] as the sub-code \tilde{C} above, we have $[\tilde{n}, \tilde{k}, \tilde{d} = \tilde{n} - 1, \tilde{\alpha} = \tilde{r}^{\tilde{k}-1}, \tilde{\beta} = \tilde{r}^{\tilde{k}-2}, \tilde{r} = \tilde{n} - \tilde{k}]$, and having $\tilde{d} = \tilde{n} - 1$ requires connecting to 1 node per block for repairs in our block model. In addition, product matrix MSR codes [3] require $\tilde{d} \geq 2\tilde{k} - 2$, and they can be used as the sub-code \tilde{C} , for which we do not necessarily have $\tilde{d} = r - 1$. We observe from (17) and (20) that the BFR-MSR point is achieved for $\tilde{k} = p + 1$, implying $k = b$, i.e. DC connects necessarily 1 node per block for data reconstruction when our Construction II gives BFR-MSR code.

2) *BFR-MBR*: To construct a BFR-MBR code, we set each sub-code \tilde{C} in Construction II as a product matrix MBR code [3], which has $\tilde{\alpha} = \tilde{d}\tilde{\beta} = \frac{2\tilde{\mathcal{M}}\tilde{d}}{\tilde{k}(2\tilde{d}-\tilde{k}+1)}$. This, together with (19), results in the following parameters of our BFR-MBR construction

$$\alpha = d\beta = \frac{2\mathcal{M}d}{k(2d - \frac{k(p+1)^2}{p^2+p+1} + p + 1)}. \quad (21)$$

From (18) and (21), we observe that the BFR-MBR point is achieved for $\tilde{k} = p + 1$.

C. Duplicated Block Design Based BFR Codes ($\rho = 0$ and $\sigma < b - 1$)

In this section, BFR codes for special case of having $\rho = 0$ is constructed. We note that $\rho = 0$ implies that DC contact all b blocks to retrieve the stored data. Before detailing the code construction, we first introduce a block design referred to as duplicated combination block design [26].

Definition 29 (DCBD). *Let $(\tilde{\kappa}, \tilde{v})$ denote the parameters for a block design, where \tilde{v} points from all possible sets of blocks each with $\tilde{\kappa}$ points. Then, duplicated combination block design (DCBD) (with repetition \tilde{r}) is a block design where the given block design is duplicated \tilde{r} times with different labeling points. (Here, total of $v = \tilde{r}\tilde{v}$ points are splitted into \tilde{r} groups of \tilde{v} points, where each group generates sub-blocks according to the given block design.)*

Example 30. DCBD with $\tilde{v} = 5$, $\tilde{\kappa} = 4$ and $\tilde{r} = 3$ is given below.

$$\begin{bmatrix} 1 & 3 & 4 & 5 & 6 & 8 & 9 & 10 & 11 & 13 & 14 & 15 \\ 1 & 2 & 4 & 5 & 6 & 7 & 9 & 10 & 11 & 12 & 14 & 15 \\ 1 & 2 & 3 & 5 & 6 & 7 & 8 & 10 & 11 & 12 & 13 & 15 \\ 1 & 2 & 3 & 4 & 6 & 7 & 8 & 9 & 11 & 12 & 13 & 14 \\ 2 & 3 & 4 & 5 & 7 & 8 & 9 & 10 & 12 & 13 & 14 & 15 \end{bmatrix} \quad (22)$$

It can be observed that each sub-block consists $\binom{\tilde{v}}{\tilde{\kappa}}$ blocks, each containing a different set of $\tilde{\kappa}$ points. Also, the same combination is repeated \tilde{r} times (with different labels for points, namely $\{6, 7, 8, 9, 10\}$ and $\{11, 12, 13, 14, 15\}$). Each row here corresponds to a block of DCBD, where sub-blocks aligned similarly in columns represent the underlying $(\tilde{\kappa}, \tilde{v})$ block design. We refer to the sub-blocks as repetition groups in the following.

Construction III (DCBD based BFR-RC): For any $(n, b, \mathcal{M}, k, \rho = 0, \alpha, d, \sigma < b - 1, \beta)$ code satisfying $b = \binom{\tilde{v}}{\tilde{\kappa}}$, $k = \frac{b}{b-1}\tilde{k}$, $d = \frac{(b-\sigma)\tilde{d}}{b-\sigma-1}$, $\alpha = (b-1)(b-\sigma)\tilde{\alpha}$, $\beta = (b-\sigma-1)(b-1)\tilde{\beta}$, $b - \sigma - 1 \mid \tilde{d}$, $b - 1 \mid \tilde{n}$, $b - 1 \mid \tilde{k}$ where $(\tilde{v}, \tilde{\kappa})$ are the underlying DCBD parameters and $[\tilde{n}, \tilde{k}, \tilde{d}, \tilde{\alpha}, \tilde{\beta}]$ represents the underlying regenerating code parameters, consider a file \mathbf{f} of size \mathcal{M} .

- Divide \mathcal{M} into $(b - \sigma)\binom{b}{b-1}$ parts of equal size $\tilde{\mathcal{M}}$, i.e., $\tilde{\mathcal{M}}(b - \sigma)b = \mathcal{M}$.
- Encode each part \mathbf{f}_i using an $[\tilde{n}, \tilde{k} = \tilde{\mathcal{M}}, \tilde{d}]$ regenerating code (referred to as the sub-code $\tilde{\mathcal{C}}$).
- Place the resulting partition codewords according to DCBD design (with $\tilde{v} = b$, $\tilde{\kappa} = b - 1$ and $\tilde{r} = b - \sigma$) such that each block has $c = \frac{\tilde{n}}{b-1}$ nodes, where each node stores $\kappa = \tilde{\kappa}\tilde{r}$ symbols, each coming from a different partition.

The system stores a file of size $\mathcal{M} = b(b - \sigma)\tilde{\mathcal{M}}$ over b blocks. We have the parameters as

$$\mathcal{M} = b(b - \sigma)\tilde{\mathcal{M}}, k = \frac{b\tilde{k}}{b-1}, d = \frac{\tilde{d}(b - \sigma)}{b - \sigma - 1}, \alpha = (b - 1)(b - \sigma)\tilde{\alpha}, \beta = (b - \sigma - 1)(b - 1)\tilde{\beta} \quad (23)$$

where we consider $\sigma < b - 1$.

The following example (with $b = 5$ and $\sigma = 2$) illustrates a repair scenario. Assume that the failed node is in the first block and it will be regenerated by blocks 2, 3 and 4 (as $b - \sigma = 3$). Considering the first repetition group, it can be observed that symbols of each of $b - \sigma = 3$ partitions ($\mathcal{P}_3, \mathcal{P}_4$ and \mathcal{P}_5) can be found in $b - \sigma - 1 = 2$ of these blocks, whereas the remaining $\sigma - 1 = 1$ partition (\mathcal{P}_1) can be found in all $b - \sigma$ blocks. (In the representation below, numbers represent the indices for partitions \mathcal{P}_i . And, the three highlighted rows for the first repetition group includes partitions $\mathcal{P}_1, \mathcal{P}_3, \mathcal{P}_4, \mathcal{P}_5$ that are relevant to the symbols stored in the block corresponding to the first row below.)

$$\begin{bmatrix} 1 & 3 & 4 & 5 & 6 & 8 & 9 & 10 & 11 & 13 & 14 & 15 \\ \mathbf{1} & 2 & \mathbf{4} & \mathbf{5} & 6 & 7 & \mathbf{9} & \mathbf{10} & \mathbf{11} & 12 & \mathbf{14} & \mathbf{15} \\ \mathbf{1} & 2 & \mathbf{3} & \mathbf{5} & \mathbf{6} & 7 & \mathbf{8} & \mathbf{10} & 11 & 12 & \mathbf{13} & \mathbf{15} \\ 1 & 2 & \mathbf{3} & \mathbf{4} & \mathbf{6} & 7 & \mathbf{8} & \mathbf{9} & \mathbf{11} & 12 & \mathbf{13} & \mathbf{14} \\ 2 & 3 & 4 & 5 & 7 & 8 & 9 & 10 & 12 & 13 & 14 & 15 \end{bmatrix} \quad (24)$$

Node Repair: Generalizing above argument, consider that one of the nodes in a block is to be repaired by contacting to $b - \sigma$ blocks. A failed node contains $\kappa = (b - 1)(b - \sigma)$ symbols, each coming from a distinct partition codeword. The properties of the underlying (DCBD) block design, (considering the first repetition group), implies that there exists $b - \sigma$ partitions of the failed node that are contained only in $b - \sigma - 1$ of the blocks contacted for repair. The remaining $b - 1 - (b - \sigma) = \sigma - 1$ partitions (of each repetition group) are contained in all of the contacted

$b - \sigma$ blocks. These partitions are referred to as the *common partitions* of a repetition group in $b - \sigma$ contacted blocks. (In the example above, partitions $\mathcal{P}_1, \mathcal{P}_6, \mathcal{P}_{11}$ are the common partitions for the first, second and third repetition group respectively.)

Using this observation for DCBD based construction, (i.e., considering all repetition groups), consider obtaining $\sigma - 1$ common partitions from $b - \sigma - 1$ blocks of each of the $\tilde{r} = b - \sigma$ repetition groups. In addition, consider obtaining remaining relevant partitions ($b - \sigma$ partitions per repetition group) from these $\tilde{r} = b - \sigma$ partition groups (total of $\tilde{r}(b - \sigma)(b - \sigma - 1)$ partitions).

These $\sigma - 1$ common partitions per repetition group over $\tilde{r} = b - \sigma$ repetition groups are contacted evenly. Namely, each other relevant partition are contacted from only $b - \sigma - 1$ blocks, the common partitions among $b - \sigma$ blocks are contacted only $b - \sigma - 1$ times (i.e., by not contacting any common point at all in only one repetition group from a block and since there are $b - \sigma$ repetition groups and $b - \sigma$ contacted blocks, we can do this process evenly for all blocks). Henceforth, from each block same amount of symbols (and same amount of symbols from each partition) is downloaded. In total, there are $(\sigma - 1)(b - \sigma)(b - \sigma - 1)$ common points and each block contributes the transmission of $(\sigma - 1)(b - \sigma - 1)$ common partitions. Hence, $\beta = [(\sigma - 1)(b - \sigma - 1) + (b - \sigma - 1)(b - \sigma)]\tilde{\beta} = (b - \sigma - 1)(b - 1)\tilde{\beta}$.

In order to have a successful regeneration for each node to be regenerated, we also require the following condition in this construction.

Lemma 31. *Construction III requires the necessary condition $\frac{\tilde{n}}{b-1} \geq \frac{\tilde{d}}{b-\sigma-1}$ for repair feasibility.*

Proof. Given $v - 1$ combinations of v points, any two combinations differs only in one point. Also, any combination is missing only one point. If one collects $b - \sigma \geq 2$ of such combinations in Construction III, then the partition with least number of instances is $b - \sigma - 1$. (This follows as the first combination is missing only one partition which is necessarily included in the second combination.) Then, by contacting any $b - \sigma$ blocks, one can recover partitions of failed node from these $b - \sigma$ blocks. (There exist at least $b - \sigma - 1$ number of blocks containing nodes storing symbols from a given partition.) Since each block has $c = \frac{\tilde{n}}{b-1}$ symbols from each partition, we require $\frac{\tilde{n}}{b-1} \geq \frac{\tilde{d}}{b-\sigma-1}$ to have repair feasibility in Construction III. \square

Therefore, a failed node can be regenerated from $d = \frac{\tilde{d}(b-\sigma)}{b-\sigma-1}$ nodes and downloading $\beta = (b - \sigma - 1)(b - 1)\tilde{\beta}$ symbols from each block. (Note that, $d_r = \frac{d}{b-\sigma} = \frac{\tilde{d}}{b-\sigma-1}$.) For each partition of the failed node, $\tilde{d}\tilde{\beta}$ symbols are downloaded, from which one can regenerate each partition. Note that, repeating combinations multiple times enables us to have uniform downloads from the nodes during repairs.

Data Collection: DC connects to $k_c = \frac{k}{b} = \frac{\tilde{k}}{b-1}$ nodes per block (as $\rho = 0$), and downloads total of $k_c\alpha$ symbols from each block. These symbols include $\frac{\tilde{k}\tilde{\alpha}}{b-1}$ symbols from each of $(b-1)(b-\sigma)$ partitions. Therefore, from all blocks, $\frac{(b-1)\tilde{k}\tilde{\alpha}}{b-1} = \tilde{k}\tilde{\alpha}$ symbols per partition is collected, from which each partition can be decoded via underlying sub-code $\tilde{\mathcal{C}}$, and the stored file \mathbf{f} can be reconstructed.

1) *BFR-MSR*: To construct a BFR-MSR code, we set each sub-code $\tilde{\mathcal{C}}$ in Construction III as an MSR code, which has $\tilde{\alpha} = \frac{\tilde{\mathcal{M}}}{k}$ and $\tilde{d}\tilde{\beta} = \frac{\tilde{\mathcal{M}}\tilde{d}}{k(\tilde{d}-k+1)}$. This, together with (23), results in the following parameters of our BFR-MSR construction

$$\alpha = (b-1)(b-\sigma)\tilde{\alpha} = \frac{\mathcal{M}}{k}, \quad (25)$$

$$d\beta = \frac{\tilde{d}(b-\sigma)}{b-\sigma-1}(b-\sigma-1)(b-1)\tilde{\beta} = \frac{\mathcal{M}d(b-\sigma-1)}{k(d(b-\sigma-1) - \frac{k(b-1)(b-\sigma)}{b} + b-\sigma)}. \quad (26)$$

From (6), we obtain that Construction III results in optimal BFR-MSR codes when $k = \frac{b}{\sigma}$ (i.e., $\tilde{k} = \frac{b-1}{\sigma}$).

2) *BFR-MBR*: To construct a BFR-MBR code, we set each sub-code $\tilde{\mathcal{C}}$ in Construction III as a product matrix MBR code [3], which has $\tilde{\alpha} = \tilde{d}\tilde{\beta} = \frac{2\tilde{\mathcal{M}}\tilde{d}}{k(2\tilde{d}-k+1)}$. This, together with (23) results in the following parameters of our BFR-MBR construction

$$\alpha = d\beta = \frac{2\mathcal{M}d(b-\sigma-1)}{k(2d(b-\sigma-1) - \frac{k(b-1)(b-\sigma)}{b} + b-\sigma)}. \quad (27)$$

From (7), we obtain that Construction III results in optimal BFR-MSR codes when $k = \frac{b^2}{b+\sigma^2-1}$ (i.e., $\tilde{k} = \frac{b(b-1)}{b+\sigma^2-1}$).

V. LOCALLY REPAIRABLE BFR CODES

A. Locality in BFR

In this section, we focus on BFR model with repair locality constraints, i.e., only a local set of blocks are available to regenerate the content of a given node. This model is suitable for both disk storage and distributed (cloud/P2P) storage systems. For example, a typical data center architecture includes multiple servers, which form racks which further form clusters, see Fig. 8. We can think of each cluster as a local group of racks where each rack contains multiple servers. Hence, in Fig. 8, we can model the data center as having b blocks (racks) where b_L blocks form a local group (cluster) and there are c nodes (servers) in each block.

In this section, we extend our study of data recovery with the block failure model to such scenarios with locality constraints. We assume that DSS maintains the data of size \mathcal{M} with at most ρ blocks being unavailable. Hence, from any $b-\rho$ blocks, data collection can be performed by contacting any k_c nodes from each of such blocks. In other words, DC can contact some set of local groups, \mathcal{B}^* , to retrieve the file stored in the DSS with $|\mathcal{B}^*| = b-\rho$. Let $\mathbf{c}_{i,j}$ denote the part of the codewords associated with i^{th} local group's j^{th} block accessed by DC, which consists of k_c nodes. Note that, we consider $k_c = c$ for full access and $k_c < c$ for partial access as before. Then, we can denote by \mathbf{c}_i the codeword seen by DC corresponding to local group \mathcal{B}_i , which has a size $j' \leq b_L$. Therefore, $\mathbf{c}_{\mathcal{B}^*} = \left\{ (\mathbf{c}_{i_1}, \dots, \mathbf{c}_{i_{|\mathcal{B}^*|}}) \right\}$ denotes the codeword corresponding to the one seen by DC, when accessing $k_c(b-\rho) = k$ nodes.

Definition 32. Let \mathbf{c} be a codeword in \mathcal{C} selected uniformly, the resilience of \mathcal{C} is defined as

$$\rho = b - \max_{\mathcal{B}^* \subseteq [b]: H(\mathbf{c}(\mathcal{B}^*)) < \mathcal{M}} |\mathcal{B}^*| - 1. \quad (28)$$

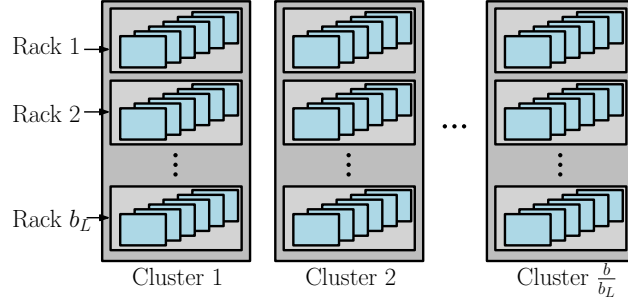


Fig. 8: Data center architecture. Each rack resembles a block and each cluster forms a local group for node repair operations in the BFR model.

We remark that the resilience ρ of a locally repairable BFR code dictates the number of block failures that the system can tolerate (analogous to the minimum distance providing the maximum number of node failures). In addition, DC can do data access by contacting any $b - \rho$ blocks.

Definition 33. Code \mathcal{C} is said to have (r_L, ρ_L) locality, if for any node j in a block i , there exists a set of blocks \mathcal{B}_i such that

- $i \in \mathcal{B}_i \subset \mathcal{B} = \{1, \dots, b\}$,
- $|\mathcal{B}_i| \leq b_L := r_L + \rho_L$,
- $\mathcal{C}|_{\mathcal{B}_i}$ is an $(n_L, b_L, K_L, k_L, \rho_L, \alpha)$ BFR code. (See Definition 1.)

Note that K_L is the size of the local data for corresponding BFR local code (i.e., \mathcal{M} in Definition 1). Such codes are denoted as BFR-LRC in the following.

1) *Upper bound on the resilience of BFR-LRC:* We provide the following bound on the resilience of BFR-LRC.

Theorem 34. The resilience of BFR-LRC can be bounded as follows.

$$\rho \leq b - \left\lceil \frac{\mathcal{M}(b_L - \rho_L)}{K_L} \right\rceil - \left(\left\lceil \frac{\mathcal{M}}{K_L} \right\rceil - 1 \right) \rho_L \quad (29)$$

Proof. The proof follows from the algorithmic approach as considered in [9], [10] and [12], and detailed in the Appendix B. \square

B. Code Constructions for Resilience Optimal BFR-LRC

In this section we propose two code constructions, which yield optimal codes in terms of resilience. Both of the constructions utilize Gabidulin coding as well as MDS codes but only the first construction uses projective plane geometry.

1) *Basic Construction with Projective Planes:*

Construction IV: Consider a file \mathbf{f} of size \mathcal{M} and projective plane of order p , $(v, \kappa, \lambda = 1) - BIBD$.

- First, encode \mathbf{f} using $[N = \frac{K_L b}{b_L}, K = \mathcal{M}, D = N - \mathcal{M} + 1]$ Gabidulin code, \mathcal{C}^{Gab} .²⁷ The resulting codeword $\mathbf{c} \in \mathcal{C}^{\text{Gab}}$ is divided into $\frac{b}{b_L}$ disjoint sets of symbols of size K_L .
- For each disjoint set, divide it into v partitions of equal size. For each partition, use $[\tilde{n}, \tilde{k} = \frac{k_L(p+1-\rho_L)}{b_L-\rho_L}]$ MDS code, where $n_L = \tilde{b}v$.
- Place the resulting encoded symbols using *Construction II* (with projective plane of order p). That is, local code in the local group will have BFR properties and constructed as in *Construction II*.

Remark 35. *Construction IV works only if $\rho_L \leq p$, since otherwise we would have non-positive value for \tilde{k} . Furthermore, we need $(b_L - \rho_L) \mid k_L(p + 1 - \rho_L)$ to have an integer \tilde{k} .*

Corollary 36. *Construction IV provides resilience-optimal codes $\mathcal{C}^{\text{BFR-LRC}}$, when $p^2 + p + 1 \mid K_L \mid \mathcal{M}$ and $b_L \mid b$.*

Proof. In order to prove that $\mathcal{C}^{\text{BFR-LRC}}$ attains the bound in (34), it is sufficient to demonstrate that any pattern of $E = b - \left\lceil \frac{\mathcal{M}(b_L - \rho_L)}{K_L} \right\rceil - \left(\left\lceil \frac{\mathcal{M}}{K_L} \right\rceil - 1 \right) \rho_L$ number of block erasures can be corrected. For $b = \frac{Nb_L}{K_L}$, we then have $E = \frac{Nb_L}{K_L} - \left\lceil \frac{\mathcal{M}(b_L - \rho_L)}{K_L} \right\rceil - \left(\left\lceil \frac{\mathcal{M}}{K_L} \right\rceil - 1 \right) \rho_L$. Also note that the worst case erasure pattern is the one when erasures happen in the smallest possible number of local groups and the number of block erasures inside a local group is the maximum possible.

Let $\mathcal{M} = \alpha_1 K_L$, then $E = b_L(\frac{N}{K_L} - \alpha_1) + \rho_L$, which corresponds to $\frac{N}{K_L} - \alpha_1$ local groups with b_L erasures and ρ_L erasures from one additional group (which results in no rank erasures for the underlying Gabidulin codewords since resilience of local group is ρ_L). Then, total rank erasures is $K_L(\frac{N}{K_L} - \alpha_1) = N - K_L \alpha_1$ which can be corrected by Gabidulin code since its minimum distance is $D = N - \mathcal{M} + 1 = N - \alpha_1 K_L + 1$. \square

Remark 37. *Although above construction yields optimal codes in terms of resilience, when $\rho_L \neq 0$, they are not rate optimal. The optimal codes should store $k_L \alpha = k_L v \tilde{\alpha} = k_L(p + 1) \tilde{\alpha}$ symbols (in each local group), since each node stores $\alpha = (p + 1) \tilde{\alpha}$ where $\tilde{\alpha}$ is the number of symbols from a partition and each node stores symbols from $p + 1$ partitions. On the other hand, with the above construction, one can store $\tilde{k} v \tilde{\alpha} = \frac{k_L(p+1-\rho_L)}{p^2+p+1-\rho_L} (p^2 + p + 1) \tilde{\alpha}$ symbols (in each local group). Note that for $\rho_L \neq 0$, $\frac{k_L(p+1-\rho_L)}{p^2+p+1-\rho_L} (p^2 + p + 1) \tilde{\alpha} \leq k_L(p + 1) \tilde{\alpha}$, which implies that the code construction is not optimal in terms rate (i.e., file size for a fixed n).*

2) *Construction for BFR-LRC with Improved Rate:* In the previous section, we utilize projective planes in code construction, however resulting codes are not optimal in terms of the rate. We now propose another construction for BFR-LRC to achieve higher rate.

Construction V: Consider a file \mathbf{f} of size \mathcal{M} .

- First, encode \mathbf{f} using $[N, K = \mathcal{M}, D = N - \mathcal{M} + 1]$ Gabidulin code, \mathcal{C}^{Gab} . The resulting codeword $\mathbf{c} \in \mathcal{C}^{\text{Gab}}$ is divided into $\frac{b}{b_L}$ disjoint groups of size $K_L = \frac{Nb_L}{b}$.
- Apply MDS codes of $[\tilde{n} = b_L c, \tilde{k} = \frac{Nb_L}{b}]$ to each disjoint group. Resulting $b_L c$ symbols are placed to each block equally (c symbols per block).

Corollary 38. *Construction V yields an optimal code, $\mathcal{C}^{\text{BFR-LRC}}$, when $b \mid Nb_L$.*

Proof. Let α_1 , β_1 and γ_1 be integers such that $\mathcal{M} = \frac{K_L}{b_L - \rho_L}(\alpha_1(b_L - \rho_L) + \beta_1) + \gamma_1$, where $1 \leq \alpha_1 \leq \frac{b}{b_L}$; $0 \leq \beta_1 \leq b_L - \rho_L - 1$; and $0 \leq \gamma_1 \leq \frac{K_L}{b_L - \rho_L} - 1$. $E = b - \left\lceil \frac{\mathcal{M}(b_L - \rho_L)}{K_L} \right\rceil - \left(\left\lceil \frac{\mathcal{M}}{K_L} \right\rceil - 1 \right) \rho_L$ is the number of block erasures that can be tolerated.

- If $\gamma_1 = \beta_1 = 0$, then $\mathcal{M} = K_L \alpha_1$. Then, we have $E = b_L(\frac{N}{K_L} - \alpha_1) + \rho_L$ number of block erasures to be tolerated, similar to Corollary 36. Thus, the worst case happens when $\frac{N}{K_L} - \alpha_1$ local groups with all of their blocks erased and one additional local group with ρ_L blocks erased. The latter does not correspond to any rank erasures since the resilience of local group is ρ_L , therefore the worst case scenario results in K_L rank erasures in each of $(\frac{N}{K_L} - \alpha_1)$ local groups. Since $D - 1 = N - \mathcal{M} = N - K_L \alpha_1$, these worst case scenarios can be corrected by the Gabidulin code.
- If $\gamma_1 = 0$ and $\beta_1 > 0$, then $\mathcal{M} = \frac{K_L}{b_L - \rho_L}(\alpha_1(b_L - \rho_L) + \beta_1)$. Hence, we have $E = b_L(\frac{N}{K_L} - \alpha_1 - 1) + b_L - \beta_1$. Thus, the worst case happens when $\frac{N}{K_L} - \alpha_1 - 1$ local groups with all of their blocks erased and one additional local group with $b_L - \beta_1$ blocks erased. Then, that scenario corresponds to $(\frac{N}{K_L} - \alpha_1)K_L - \beta_1 \frac{K_L}{b_L - \rho_L}$ rank erasures, which can be corrected by Gabidulin code since $D - 1 = N - \mathcal{M} = N - \alpha_1 K_L - \beta_1 \frac{K_L}{b_L - \rho_L}$.
- If $\gamma_1 > 0$, then $\mathcal{M} = \frac{K_L}{b_L - \rho_L}(\alpha_1(b_L - \rho_L) + \beta_1) + \gamma_1$. Hence, we have $E = b_L(\frac{N}{K_L} - \alpha_1 - 1) + b_L - \beta_1 - 1$. Thus, the worst case happens when $\frac{N}{K_L} - \alpha_1 - 1$ local groups with all of their blocks erased and one additional local group with $b_L - \beta_1 - 1$ blocks erased. Then, that scenario corresponds to $(\frac{N}{K_L} - \alpha_1)K_L - (\beta_1 + 1) \frac{K_L}{b_L - \rho_L}$ rank erasures, which can be corrected by Gabidulin code since $D - 1 = N - \mathcal{M} = N - \alpha_1 K_L - \beta_1 \frac{K_L}{b_L - \rho_L} - \gamma_1$ and $\gamma_1 < \frac{K_L}{b_L - \rho_L}$.

□

Remark 39. *We note that the above construction is similar to that of [12], here modified for the block failure model to achieve resilience optimal construction with the improved rate.*

C. Local Regeneration for BFR Codes

In the previous section, MDS codes are utilized in local groups. These codewords, however, are not repair bandwidth optimal. As an alternative, regenerating codes can be used in local groups to construct codes which have better trade-off in terms of repair bandwidth. Differentiating between the two important points, we denote BFR-LRC codes with regenerating code properties which operate at minimum per-node storage point as BFR-MSR-LRC. Similarly, the codes operating at minimum repair bandwidth point are called BFR-MBR-LRC.

Let $\mathcal{G}_1, \dots, \mathcal{G}_{\frac{b}{b_L}}$ represent the disjoint set of indices of blocks where \mathcal{G}_i represents local group i , which has b_L blocks. A failed node in one of the blocks in \mathcal{G}_i is repaired by contacting any $b_L - \sigma_L$ blocks within the group. A newcomer downloads β symbols from $\frac{d_L}{b_L - \sigma_L}$ nodes from each of $b_L - \sigma_L$ blocks. That is, the local group has the properties of repair bandwidth efficient BFR codes as studied in Section III.

Definition 40 (Uniform rank accumulation (URA) codes). *Let G be a generator matrix for a BFR code \mathcal{C} . Columns of G produce the codewords, henceforth we can think of each α columns (also referred to as thick column) of G as a representation of a node storing α symbols. Then, a block can be represented by c such thick columns. Let S_i be an arbitrary subset of i such blocks. \mathcal{C} is an URA code, if the restriction $G|_{S_i}$ of G to S_i , has rank ρ_i that is independent of specific subset S_i and it only depends on $|S_i|$.*

Remark 41. *Generally, URA codes are associated with rank accumulation profile to calculate the rank of any subset S_i . However, rank accumulation profile is not required but rather rank accumulation is enough for a code to be considered as URA code. We note that for BFR-MSR/MBR codes, we do not have a specific rank accumulation profile but that does not rule out BFR-MSR/MBR codes being URA since they still obey the rank accumulation property. Specifically, we note that $H(b_{S_i}) = f(|S_i|)$ for both BFR-MSR/MBR, which makes them URA codes.*

Following similar steps as introduced in [16], resilience upper bound can be derived when local codes are URA codes. Consider the finite length vector (b_1, \dots, b_{b_L}) and its extension with b_L period as $b_{i+jb_L} = b_i$, $1 \leq i \leq b_L$, $j \geq 1$. Let $H(b_S)$ denote the entropy of set of blocks \mathcal{S} ,

$$H(b_S) = \sum_{i=1}^{|S|} H(b_{j_i} | b_{j_1}, \dots, b_{j_{i-1}}), \quad |S| \geq 1, \quad \mathcal{S} = \{j_1, \dots, j_{|S|}\} \subseteq [b]. \quad (30)$$

$H(\cdot)$ function in the above only depends on the structure of \mathcal{S} , which contains μ number of local groups each with b_L blocks and additional set of ϕ blocks. We denote the entropy of this set by referring to this structure and use $H(\cdot)$ (with some abuse of notation) function with an argument of size of \mathcal{S} . More specifically, for integers $\mu \geq 0$ and $1 \leq \phi \leq b_L$, let $H(\mu b_L + \phi) = \mu K_L + H(\phi)$. (Note that, due to URA property, entropy is only a function of number of blocks here.) For the inverse function $H^{(inv)}$, we set $H^{(inv)}(\varphi)$ for $\varphi \geq 1$, to be largest integer \mathcal{S} such that $H(b_S) \geq \varphi$. Then, we have for $\tilde{\mu} \geq 0$ and $1 \leq \tilde{\phi} \leq K_L$, $H^{(inv)}(\tilde{\mu} K_L + \tilde{\phi}) = \tilde{\mu} b_L + H^{(inv)}(\tilde{\phi})$, where $H^{(inv)}(\tilde{\phi}) \leq \min\{b - \rho_L, b - \sigma_L\}$.

Theorem 42. *The resilience of BFR-LRC is upper bounded by $\rho \leq b - H^{(inv)}(\mathcal{M})$.*

When URA codes are used as local codes, we have the file size bound for resilience optimal codes as $\mathcal{M} \leq H(b - \rho) = \mu K_L + H(\phi)$, where $\mu = \left\lfloor \frac{b-\rho}{b_L} \right\rfloor$ and $\phi = b - \rho - \mu b_L$. Note that if $\phi \geq \min\{b_L - \rho_L, b_L - \sigma_L\}$, then $H(\sum_{i=1}^{\phi} b_i) = K_L$ since from any such ϕ blocks, one can regenerate all symbols or retrieve the content stored in the corresponding local group. Therefore, the case of having $\phi \geq \min\{b_L - \rho_L, b_L - \sigma_L\}$ results in $\mathcal{M} = (\mu + 1)K_L$ and we will mainly focus on the otherwise in the following.

1) *Local Regeneration with BFR-MSR Codes:* At first, we analyze the case where BFR-MSR codes are used inside local groups to have better trade-off in terms of repair bandwidth. When

BFR-MSR codes are used, dimension of local code is given by

$$K_L = \sum_{i=1}^{b_L} H(b_i | b_1, \dots, b_{i-1}) = k_L \alpha. \quad (31)$$

Since BFR-MSR is a class of URA codes, we can upper bound the resilience of BFR-MSR-LRC as $\rho \leq b - H^{(\text{inv})}(\mathcal{M})$, where for BFR-MSR codes we have

$$H^{(\text{inv})}(\mu K_L + \phi) = \mu b_L + \phi, \quad (32)$$

for some $\mu \geq 0$ and $1 \leq \phi \leq K_L$ and ϕ is determined from $\frac{K_L(\varphi-1)}{b_L-\rho_L} < \phi \leq \frac{K_L\varphi}{b_L-\rho_L}$. Then, we can derive the file size bound for optimal BFR-MSR-LRC as $\mathcal{M} \leq H(b-\rho)$, i.e.;

$$\mathcal{M} \leq \mu K_L + \frac{K_L \varphi}{b_L - \rho_L}, \quad (33)$$

where $\mu = \left\lfloor \frac{b-\rho}{b_L} \right\rfloor$ and $\varphi = b - \rho - \mu b_L < \min \{b_L - \rho_L, b_L - \sigma_L\}$. If $\varphi \geq \min \{b_L - \rho_L, b_L - \sigma_L\}$, then $\phi = K_L$ and $\mathcal{M} \leq K_L(\mu + 1)$.

2) *Local Regeneration with BFR-MBR Codes:* In the following, we focus on the case where the local groups form BFR-MBR codes. Then, the dimension of the local code is given by

$$\begin{aligned} K_L &= \sum_{i=1}^{b_L} H(b_i | b_1, \dots, b_{i-1}) \\ &= \begin{cases} \beta(k_L d_L - \frac{k_L^2(b_L - \rho_L - 1)}{2(b_L - \rho_L)}), & \text{if } \frac{d_L}{b_L - \sigma_L} \geq \frac{k_L}{b_L - \rho_L} \text{ and } \sigma_L \leq \rho_L \\ \beta(k_L d_L - \frac{k_L^2(b_L - \sigma_L)(b_L + \sigma_L - 2\rho_L - 1)}{2(b_L - \rho_L)^2}), & \text{if } \frac{d_L}{b_L - \sigma_L} \geq \frac{k_L}{b_L - \rho_L} \text{ and } \sigma_L > \rho_L \\ \beta(\frac{k_L d_L(\rho_L - \sigma_L + 1)}{b_L - \sigma_L} + \frac{d_L^2(b_L - \rho_L)(b_L - \rho_L - 1)}{2(b_L - \sigma_L)^2}), & \text{if } \frac{d_L}{b_L - \sigma_L} < \frac{k_L}{b_L - \rho_L} \text{ and } \sigma_L < \rho_L \end{cases} \end{aligned} \quad (34)$$

Using the fact that BFR-MBR is URA code, the upper bound on the resilience of BFR-MBR-LRC is given by $\rho \leq b - H^{(\text{inv})}(\mathcal{M})$, where for BFR-MBR codes we have

$$H^{(\text{inv})}(\mu K_L + \phi) = \mu b_L + \phi, \quad (35)$$

for some $\mu \geq 0$ and $1 \leq \phi \leq K_L$ and φ is determined from

$$\varphi = \begin{cases} \beta(\frac{k_L d_L(\varphi-1)}{b_L - \rho_L} - \frac{k_L^2(\varphi-2)(\varphi-1)}{2(b_L - \rho_L)^2}) < \phi \leq \beta(\frac{k_L d_L \varphi}{b_L - \rho_L} - \frac{k_L^2 \varphi(\varphi-1)}{2(b_L - \rho_L)^2}), & \text{if } \frac{d_L}{b_L - \sigma_L} \geq \frac{k_L}{b_L - \rho_L} \\ \frac{d_L(\varphi-1)\beta}{b_L - \sigma_L} (\frac{d_L}{2} + \frac{(2k_c - d_r)(b_L - \sigma_L - \varphi + 2)}{2}) < \phi \leq \frac{d_L \varphi \beta}{b_L - \sigma_L} (\frac{d_L}{2} + \frac{(2k_c - d_r)(b_L - \sigma_L - \varphi + 1)}{2}), & \text{o.w} \end{cases} \quad (36)$$

where $k_c = \frac{k_L}{b_L - \rho_L}$ and $d_r = \frac{d_L}{b_L - \sigma_L}$. Now, the file size bound for an optimal BFR-MBR-LRC is given by

$$\mathcal{M} \leq \mu K_L + \Delta, \quad (37)$$

where $\mu = \left\lfloor \frac{b-\rho}{b_L} \right\rfloor$, $\varphi = b - \rho - \mu b_L < \min \{b_L - \rho_L, b_L - \sigma_L\}$ and

$$\Delta = \begin{cases} \beta(\frac{k_L d_L \varphi}{b_L - \rho_L} - \frac{k_L^2 \varphi(\varphi-1)}{2(b_L - \rho_L)^2}), & \text{if } \frac{d_L}{b_L - \sigma_L} \geq \frac{k_L}{b_L - \rho_L} \\ \beta(\frac{k_L d_L \varphi}{b_L - \rho_L} (\frac{b_L - \sigma_L - \varphi + 1}{b_L - \sigma_L}) + \frac{d_L^2 \varphi(\varphi-1)}{2(b_L - \sigma_L)^2}), & \text{o.w} \end{cases} \quad (38)$$

If $\varphi \geq \min \{b_L - \rho_L, b_L - \sigma_L\}$, then $\mathcal{M} \leq K_L(\mu + 1)$.

3) *Construction of BFR-MSR/MBR-LRC*: In order to construct codes for BFR-MSR/MBR-LRC, we utilize our earlier construction that is based on Duplicated Combination Block Designs.

Construction VI: Consider a file \mathbf{f} of size \mathcal{M} .

- First, encode \mathbf{f} using $[N, K = \mathcal{M}, D = N - \mathcal{M} + 1]$ Gabidulin code, \mathcal{C}^{Gab} . The resulting codeword $\mathbf{c} \in \mathcal{C}^{\text{Gab}}$ is divided into $\frac{b}{b_L}$ disjoint local groups of size $\frac{Nb_L}{b}$.
- Apply *Construction III* to each local group.

Remark 43. *Construction VI requires $\frac{N}{b(b_L - \sigma_L)}$ to be integer since in each local group we utilize Construction III, where a sub-code (regenerating code) is used with $\tilde{k} = \tilde{\mathcal{M}}$ and $\tilde{\mathcal{M}}$ is obtained by partitioning local file into $(b_L - \sigma_L)b_L$ parts. Furthermore, due to use of Construction III, Construction VI results in codes with $\rho_L = 0$. (This implies that code can tolerate any ρ block erasures, but the content of the local group will be reduced even after a single block erasure in the local group.)*

Corollary 44. *Construction VI yields an optimal code, with respect to resilience, when $b(b_L - \sigma_L) \mid N$ and $K_L \mid \mathcal{M}$.*

Proof. We need to show that $E = b - \left\lceil \frac{Mb_L}{K_L} \right\rceil$ number of block erasures are tolerated since $\rho_L = 0$. If $K_L \mid \mathcal{M}$, then $\mathcal{M} = \alpha_1 K_L$ and $E = b - \alpha_1 b_L$, which implies that the worst case erasure scenario is having $\frac{N}{K_L} - \alpha_1$ local groups with b_L erasures. Then, total rank erasure is $K_L(\frac{N}{K_L} - \alpha_1) = N - K_L \alpha_1$, which can be corrected by Gabidulin code since $D - 1 = N - \mathcal{M} = N - \alpha_1 K_L$. \square

We will utilize *Construction VI* to obtain optimal BFR-MSR-LRC. *Construction VI* can be used to construct BFR-MBR-LRC as well, if BFR-MBR codes are used inside local groups.

Corollary 45. *Construction VI yields an optimal LRC with respect to file size bounds when $\frac{N}{b} \mid \mathcal{M}$ for BFR-MSR-LRC.*

Proof. If $\frac{N}{b} \mid \mathcal{M}$, then $\frac{K_L}{b_L - \rho_L} \mid \mathcal{M}$ (since $K_L = \frac{Nb_L}{b}$) for $\rho_L = 0$, which is the case. Therefore, we can write $\mathcal{M} = \frac{K_L}{b_L}(\alpha_1 b_L + \beta_1)$ for $0 \leq \alpha_1 \leq \frac{b}{b_L}$ and $0 \leq \beta_1 \leq b_L - \sigma_L$.

- If $\beta_1 = 0$, then $b - \rho = \left\lceil \frac{Mb_L}{K_L} \right\rceil = \alpha_1 b_L$. Therefore $\mu = \left\lfloor \frac{b - \rho}{b_L} \right\rfloor = \alpha_1$ and $\varphi = b - \rho - \mu b_L = 0$, which implies that $\mu K_L + \frac{K_L \varphi}{b_L} = \alpha_1 K_L = \mathcal{M}$.
- If $\beta_1 \neq 0$, then $b - \rho = \left\lceil \frac{Mb_L}{K_L} \right\rceil = \alpha_1 b_L + \beta_1$. Therefore $\mu = \left\lfloor \frac{b - \rho}{b_L} \right\rfloor = \alpha_1$ and $\varphi = b - \rho - \mu b_L = \beta_1$, which means $\mu K_L + \frac{K_L \varphi}{b_L} = \alpha_1 K_L + \frac{K_L \beta_1}{b_L} = \mathcal{M}$.

\square

VI. DISCUSSION

A. Repair Delay

In this section, we analyze the repair delay by considering the available bandwidth between the blocks. (In DSS architectures racks are connected through switches, referred to as top-of-rack (TOR) switches, and the model here corresponds to the communication delay between these

switches.) Since in schemes that use BFR, a failed node requires $d_r\beta_{\text{BFR}}$ amount of data from each of $b - \sigma$ blocks, the repair delay (normalized with file size \mathcal{M}) can be calculated as follows.

$$RT_{\text{BFR}} = \max_{i \in \mathcal{B}_h} \frac{d_r\beta_{\text{BFR}}}{\mathcal{M}BW_i}, \quad \mathcal{B}_h = \{1, \dots, b - \sigma\}, \quad (39)$$

where BW_i is the bandwidth for block i and \mathcal{B}_h is the set of blocks that help in the repairs. We assume the repairs through blocks are performed in parallel. Throughout this section, we assume that all bandwidths are identical ($BW_i = BW, \forall i$), hence the repair time of a failed node is given by $\frac{d_r\beta_{\text{BFR}}}{\mathcal{M}BW}$.

In an identical setting, we can also analyze the repair delay of regenerating codes. Note that regenerating codes do not require any symmetric distribution of helper nodes among blocks. Hence repair delay of regenerating codes is,

$$RT_{\text{RC}}(s) = \max_i \frac{d_i\beta_{\text{RC}}}{\mathcal{M}BW}, \quad s \in \mathcal{S} = \left\{ \{d_1, \dots, d_{|\mathcal{B}|}\} \text{ s.t. } \sum_i d_i = d, \mathcal{B} = \{1, \dots, b\} \right\} \quad (40)$$

where a system s refers to a selection of d_i (the number of helper nodes in block i) such that the sum of helper nodes is equal to d . In other words, repair delay will be affected by the block with the most helper node. Furthermore, the average repair delay can be calculated as

$$\mathbb{E}[RT_{\text{RC}}(s)] = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} RT_{\text{RC}}(s) \quad (41)$$

We may encounter some d values, which are not attainable in the BFR model since for BFR codes we require $d \leq n - c$ because of the assumption that a failed node does not connect to any nodes in the same block. Furthermore, to compare the codes in a fair manner, we assume that regenerating codes connect to any d nodes from $b - \sigma$ blocks. Henceforth, in our comparisons, we instead calculate repair delay for regenerating codes as

$$RT_{\text{RC}}(s) = \max_i \frac{d_i\beta_{\text{RC}}}{\mathcal{M}BW}, \quad s \in \mathcal{S} = \left\{ \{d_1, \dots, d_{|\mathcal{B}_h|}\} \text{ s.t. } \sum_i d_i = d, \mathcal{B}_h = \{1, \dots, b - \sigma\} \right\} \quad (42)$$

where the helper nodes are chosen from $b - \sigma$ blocks but the number of helpers in each block is not necessarily the same, only requirement is to have the sum of the helpers equal to d . Average repair time can be calculated similar to (41) where the difference being the system s satisfies the condition that helpers are chosen from $b - \sigma$ blocks.

One can allow symmetric distribution of regenerating codes among $d - \sigma$ blocks as well, similar to BFR codes. We'll denote these codes by MSR-SYM or MBR-SYM in the following.

We examine the case where $b = 7$ and $n = 21$, which means each block contains $c = 3$ nodes. We also set $\sigma = 3$ so that $b - \sigma = 4$. Note that since we are comparing relative values, the values we assign to BW and \mathcal{M} does not change the result as long as they are same across all comparisons. In other words, one can think our results as normalized repair delays. At first, we find out all possible ρ, d_r, k_c, d and k values accordingly from Section III. After identifying all possible parameter sets, we calculate repair times for both BFR-MSR and BFR-MBR codes. For

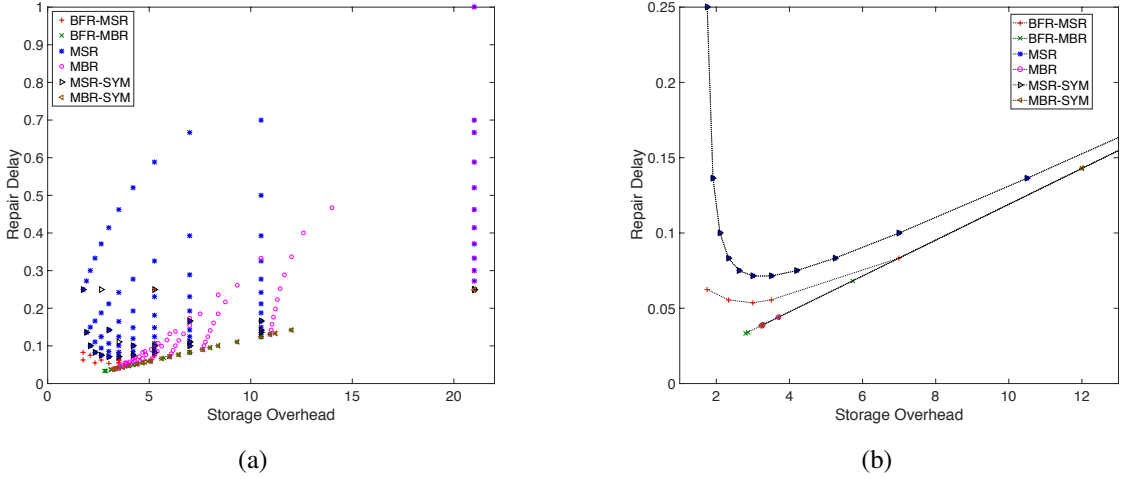


Fig. 9: Repair delay vs. storage overhead comparisons for $b = 7$, $n = 21$ and $\sigma = 3$. (a) Data points for possible cases of each node. (b) Lower envelope of Fig. 9a when zoomed in.

regenerating codes, we choose all possible parameter sets as long as $d \leq n - \sigma c$ since we impose regenerating codes to perform repairs from $b - \sigma$ blocks. For each parameter set, we calculate repair times of all possible helper node allocations and then calculate the mean of those. The mean is reported for each data point in Fig. 9 for one set of parameters for both MSR and MBR codes. Finally, we allow symmetric distribution of helper nodes among $b - \sigma$ blocks, and report MSR-SYM and MBR-SYM.

Repair delay vs. storage overhead results are depicted on Fig. 9. In Fig. 9a we indicate all data points, whereas in Fig. 9b, lower envelope of Fig. 9a for storage overheads less than 13. As expected, one can observe that there are more data points for regenerating codes than BFR, since BFR requires $b - \sigma \mid d$, which limits the number of possible sets of parameters for BFR. First, lower envelopes of MSR and MSR-SYM are the best repair times for MSR which are achieved when d gets its highest possible value, $(b - \sigma)c$. In other words, all nodes are utilized hence there is no different possible connection schemes for MSR. Interestingly, given storage overhead, we observe that in some cases MSR codes perform better than MBR codes even though MBR codes minimize repair bandwidth. On the other hand, when distributed symmetrically across blocks, MBR-SYM outperforms both MBR and MSR-SYM in all cases. When we compare BFR-MSR and BFR-MBR, we can observe that BFR-MBR has lower repair delay for all cases but still they perform the same when storage overhead gets larger. Furthermore, we observe that unlike MBR-MSR comparison, BFR performs more regularly, meaning BFR-MBR is better than BFR-MSR always. Next, if we compare all schemes, we observe that convex hulls of BFR-MBR, MBR and MBR-SYM follows the same line. Note that repair delay of BFR-MSR is below MSR and it performs the same as storage overhead increases. Also, BFR-MBR outperforms MSR-SYM and performs identical to MBR-SYM. For lower storage overheads, we observe that BFR codes operate well (both BFR-MSR and BFR-MBR) whereas existing RC or RC-SYM codes (MBR and MBR-SYM codes do not even exist for overhead below 3.23) performs worse than BFR.

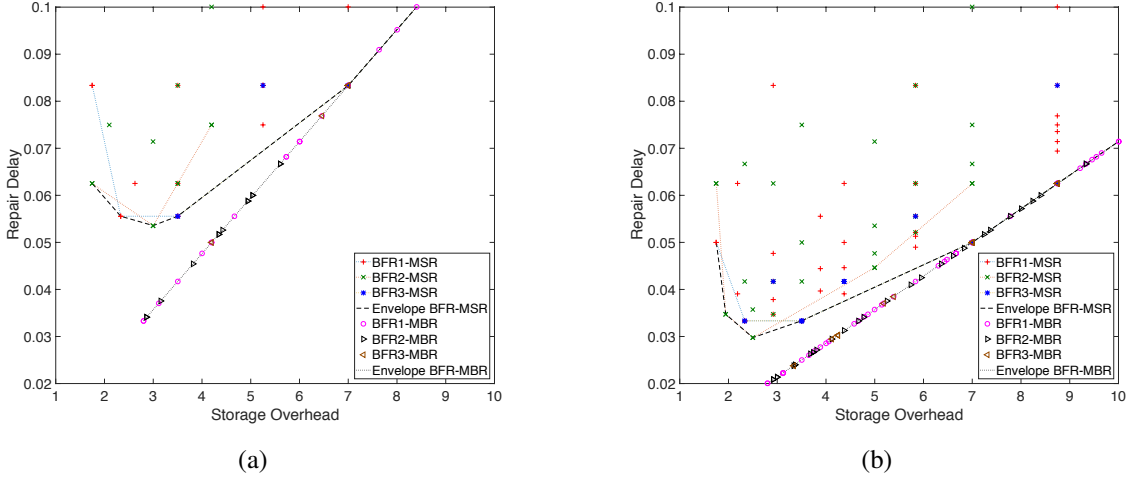


Fig. 10: Repair delay vs. storage overhead comparisons for $b = 7$ and $\sigma = 3$, (a) when $n = 21$, (b) when $n = 35$.

Finally, note that within BFR schemes, we may encounter different α and β values depending on the parameters. Differentiating between cases, let BFR1 denote the schemes with $d_r \geq k_c$ and $\rho \geq \sigma$, BFR2 denote $d_r \geq k_c$ and $\rho < \sigma$, and BFR3 denote $d_r < k_c$ and $\rho \geq \sigma$. The resulting β values for these schemes may not be the same for same k and d . In Fig. 10, we examine these BFR schemes (for storage overhead less than 10). We observe that the convex hull for BFR-MBR is a line and all BFR-MBR schemes operate on that line. Furthermore, different MSR schemes can perform better depending on the storage overhead.

B. Relaxed BFR

It's challenging to find general constructions that allow for repair and data collection schemes to operate with *any* ρ and σ blocks with *any* subset of d_r or k_c . Hence, in this section, we relax “any” requirement of BFR to table-based recovery schemes that guarantee the existence of helper/data recovery nodes for every possible repair/data collection scenarios. By altering this, we are able to use another combinatorial design and obtain relaxed BFR codes (R-BFR) for wider set of parameters.³

1) Resolvable Balanced Incomplete Block Design (RBIBD):

Definition 46. A parallel class is the set of blocks that partition the point set. A resolvable balanced incomplete block design is a $(v, \kappa, \lambda) - BIBD$ whose blocks can be partitioned into parallel classes.

³We note that under this setting the fundamental limits will be different than BFR codes. The focus here is more on operating at BFR performance under a relaxation of “any” requirement.

An example of $(9, 3, 1) - RBIBD$ is given below, where each column (consisting of 9 points in 3 blocks) forms a parallel class.

$$\begin{array}{cccc} \{1, 2, 3\} & \{1, 4, 7\} & \{1, 5, 9\} & \{1, 6, 8\} \\ \{4, 5, 6\} & \{2, 5, 8\} & \{2, 6, 7\} & \{2, 4, 9\} \\ \{7, 8, 9\} & \{3, 6, 9\} & \{3, 4, 8\} & \{3, 5, 7\} \end{array} \quad (43)$$

2) *R-BFR-RC with RBIBD*: In this section, we show that RBIBDs can be used to construct R-BFR codes for any $\rho \geq 0$ and $\sigma \geq 1$. Considering RBIBDs (with $\lambda = 1$) as defined above, we construct blocks each containing the same number of nodes that store symbols belonging to different partitions. For instance, utilizing (43), a block can be formed to contain 12 nodes, where 4 nodes store symbols in the form of $\{1, 2, 3\}$, (referred to as “type” below), 4 nodes store symbols of the type $\{4, 5, 6\}$ and the other 4 nodes store symbols of the type $\{7, 8, 9\}$. We refer to blocks of the same type as sub-block. Assume that one of the nodes that store the symbols of the type $\{1, 2, 3\}$ is failed. A newcomer may download symbols from any other subset of blocks but instead of connecting to any d_r nodes from each block, we consider connecting to any $\frac{d_r}{3}$ nodes of type $\{1, 4, 7\}$, any $\frac{d_r}{3}$ nodes of type $\{2, 5, 8\}$ and any $\frac{d_r}{3}$ nodes of type $\{3, 6, 9\}$ in the second block and so on.⁴ Similarly, DC can connect to any $\frac{k_c}{3}$ from each sub-blocks. Therefore, the requirement of *any* set of nodes from a block is changed to *any* subset of nodes from a *sub-block*. Note that, RBIBD still preserves any ρ and σ properties.

In the general case of any ρ and σ , we still have the same relationship as before, c.f., (44), since repair property with any $b - \sigma$ blocks or DC property with any $b - \rho$ blocks does not change this relationship but instead it only alters which d_r and k_c nodes that are connected in a block, where $d_r = \frac{d}{b - \sigma}$, $k_c = \frac{k}{b - \rho}$. Also, to ensure the repair and DC properties, $c = \frac{n}{b} \geq \max\{k_c, d_r\}$ must be satisfied.

$$M = v\tilde{M}, k = \frac{v}{\kappa}\tilde{k}, d = \kappa\tilde{d}, \alpha = \kappa\tilde{\alpha}, \beta = \tilde{\beta}. \quad (44)$$

With this construction, the same steps provided for BFR-MSR and BFR-MBR cases in the previous section can be followed. In the general case, we have three cases depending on the values of d_r , k_c , σ and ρ and the corresponding cut values can be found using Theorems 13, 21, and Lemma 18. Solving for these minimum storage and bandwidth points, optimal \tilde{k} values can be found. We provide these result in the following subsections.

3) *R-BFR-MSR*: To construct a R-BFR-MSR code, we set each sub-code \tilde{C} as an MSR code which has $\tilde{\alpha} = \frac{\tilde{M}}{\tilde{k}}$ and $\tilde{d}\tilde{\beta} = \frac{\tilde{M}\tilde{d}}{\tilde{k}(\tilde{d} - \tilde{k} + 1)}$. This together with (44) results in the following parameters for R-BFR-MSR construction

$$\alpha = \tilde{\alpha}\kappa = \frac{\mathcal{M}}{k} \quad d\beta = \tilde{d}\kappa\tilde{\beta} = \frac{\mathcal{M}d}{k(d - \frac{k\kappa^2}{v} + \kappa)}. \quad (45)$$

⁴This construction can be viewed as a generalization of table-based repair for regenerating codes, see e.g., fractional regenerating codes proposed in [25].

Using the relationships above together with (3), (6) and (9) we obtain the optimal value as follows: (note that $b = \frac{v-1}{\kappa-1}$)

$$\tilde{k} = \begin{cases} \frac{\kappa^2(b-\rho)}{(\kappa^2-v)(b-\rho)+v}, & d_r \geq k_c \text{ and } \sigma \leq \rho \\ \frac{\kappa^2(b-\rho)}{\kappa^2(b-\rho)-v(b-\sigma)}, & d_r \geq k_c \text{ and } \sigma < \rho \\ \frac{d(b-\rho-1)+\kappa(b-\sigma)}{\kappa(b-\sigma)}, & d_r < k_c \text{ and } \sigma \leq \rho. \end{cases} \quad (46)$$

4) *R-BFR-MBR*: To construct R-BFR-MBR code, we set each sub-code \tilde{C} as an MBR code which has $\tilde{\alpha} = \tilde{d}\tilde{\beta} = \frac{2\mathcal{M}\tilde{d}}{k(2\tilde{d}-\tilde{k}+1)}$. This together with (44) results in the following parameters for our R-BFR-MBR construction

$$\alpha = d\beta = \frac{2\mathcal{M}d}{k(2d - \frac{k\kappa^2}{v} + \kappa)}. \quad (47)$$

We can solve for optimal value in MBR case using the equations above together with (4), (7) and (10). Resulting optimal values are as follows (note that $b = \frac{v-1}{\kappa-1}$)

$$\tilde{k} = \begin{cases} \frac{\kappa^2(b-\rho)}{(\kappa^2-v)(b-\rho)+v}, & d_r \geq k_c \text{ and } \sigma \leq \rho \\ \frac{\kappa^2(b-\rho)^2}{\kappa^2(b-\rho)^2-v(b-\sigma)(b+\sigma-2\rho-1)}, & d_r \geq k_c \text{ and } \sigma < \rho \\ \frac{\frac{2d(b-\rho-1)}{\kappa} - b + \sigma \pm \sqrt{(\frac{2d(b-\rho+1)}{\kappa} - b + \sigma)^2 - \frac{4d^2(b-\rho)(b-\rho-1)}{v}}}{2(b-\sigma)}, & d_r < k_c \text{ and } \sigma \leq \rho. \end{cases} \quad (48)$$

VII. CONCLUSION

We introduced the framework of block failure resilient (BFR) codes that can recover data stored in the system from a subset of available blocks with a load balancing property. Repairability is then studied, file size bounds are derived, BFR-MSR and BFR-MBR points are characterized, explicit code constructions for a wide parameter settings are provided for limited range of σ and ρ . We then analyzed BFR for a broader range of parameters and characterized the file size bounds in these settings and also proposed code constructions achieving the critical points on the trade-off curve. Locally repairable BFR codes are studied where the upper bound on the resilience of DSS is characterized and two-step encoding process is proposed to achieve optimality. We finally analyzed the repair delay of BFR codes and compare those with regenerating codes, and provide constructions with table-based repair and data recovery properties.

Constructions reported here are based on combinatorial designs, which necessitate certain parameter sets. As a future work, optimal code constructions for the BFR model can be studied further (especially for $\rho > 0$ case). Also, the file size bound for the case of having $d_r \geq k_c$ and $\sigma > \rho$ is established here, and we conjecture that this expression corresponds to the min-cut. The proof for this conjecture resisted our efforts thus far. Furthermore, repair delay with a uniform BW assumption is studied here. Different bandwidth and more realistic communication schemes (e.g., queuing models) can be studied. Further, system implementations can be performed and more realistic analysis can be made for disk storage and distributed (cloud/P2P) storage scenarios.

REFERENCES

- [1] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4539–4551, Sep. 2010.
- [2] I. Tamo, Z. Wang, and J. Bruck, "Zigzag codes: MDS array codes with optimal rebuilding," *IEEE Trans. Inf. Theory*, vol. 59, no. 3, pp. 1597–1616, Mar. 2013.
- [3] K. V. Rashmi, N. B. Shah, and P. V. Kumar, "Optimal exact-regenerating codes for distributed storage at the MSR and MBR points via a product-matrix construction," *IEEE Trans. Inf. Theory*, vol. 57, no. 8, pp. 5227–5239, Aug. 2011.
- [4] D. Papailiopoulos, A. Dimakis, and V. Cadambe, "Repair optimal erasure codes through Hadamard designs," *IEEE Trans. Inf. Theory*, vol. 59, no. 5, pp. 3021–3037, May 2013.
- [5] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, "A survey on network codes for distributed storage," *Proc. IEEE*, vol. 99, no. 3, pp. 476–489, Mar. 2011.
- [6] Y. Wu, R. Dimakis, and K. Ramchandran, "Deterministic regenerating codes for distributed storage," in *Proc. 45th Annual Allerton Conference on communication, control and computing*, Monticello, IL, Sep. 2007.
- [7] Y. Wu, "Existence and construction of capacity-achieving network codes for distributed storage," *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 2, pp. 277–288, Feb. 2010.
- [8] V. Cadambe, C. Huang, and J. Li, "Permutation code: Optimal exact-repair of a single failed node in MDS code based distributed storage systems," in *Proc. 2011 IEEE International Symposium on Information Theory (ISIT 2011)*, Saint Petersburg, Russia, Jul. 2011.
- [9] P. Gopalan, C. Huang, H. Simitci, and S. Yekhanin, "On the locality of codeword symbols," *IEEE Trans. Inf. Theory*, vol. 58, no. 11, pp. 6925–6934, Nov. 2012.
- [10] D. S. Papailiopoulos and A. G. Dimakis, "Locally repairable codes," in *Proc. 2012 IEEE International Symposium on Information Theory (ISIT 2012)*, Boston, MA, Jul. 2012.
- [11] F. Oggier and A. Datta, "Self-repairing homomorphic codes for distributed storage systems," in *Proc. 2011 IEEE INFOCOM*, Shanghai, China, Apr. 2011.
- [12] A. S. Rawat, O. O. Koyluoglu, N. Silberstein, and S. Vishwanath, "Optimal locally repairable and secure codes for distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 60, no. 1, pp. 212–236, Jan. 2014.
- [13] N. Prakash, G. Kamath, V. Lalitha, and P. Kumar, "Optimal linear codes with a local-error-correction property," in *Proc. 2012 IEEE International Symposium on Information Theory (ISIT 2012)*, Boston, MA, Jul. 2012.
- [14] C. Huang, M. Chen, and J. Li, "Pyramid codes: Flexible schemes to trade space for access efficiency in reliable data storage systems," in *Proc. 2007 Sixth IEEE International Symposium on Network Computing and Applications (IEEE NCA07)*, Cambridge, MA, Jul. 2007.
- [15] G. M. Kamath, N. Prakash, V. Lalitha, and P. V. Kumar, "Codes with local regeneration," *CoRR*, vol. abs/1211.1932, Nov. 2012.
- [16] G. M. Kamath, N. Silberstein, N. Prakash, A. S. Rawat, V. Lalitha, O. O. Koyluoglu, P. V. Kumar, and S. Vishwanath, "Explicit MBR all-symbol locality codes," in *Proc. 2013 IEEE International Symposium on Information Theory (ISIT 2013)*, Istanbul, Turkey, Jul. 2013.
- [17] A. Wang and Z. Zhang, "Repair locality with multiple erasure tolerance," *IEEE Trans. Inf. Theory*, vol. 60, no. 11, pp. 6979–6987, Nov. 2014.
- [18] A. S. Rawat, D. S. Papailiopoulos, A. G. Dimakis, and S. Vishwanath, "Locality and availability in distributed storage," *CoRR*, vol. abs/1402.2011, Feb. 2014.
- [19] M. Sathiamoorthy, M. Asteris, D. Papailiopoulos, A. G. Dimakis, R. Vadali, S. Chen, and D. Borthakur, "XORing elephants: Novel erasure codes for big data," *Proc. VLDB Endow.*, vol. 6, no. 5, pp. 325–336, Mar. 2013.
- [20] C. Huang, H. Simitci, Y. Xu, A. Ogus, B. Calder, P. Gopalan, J. Li, and S. Yekhanin, "Erasure coding in Windows azure storage," in *Proc. USENIX Annual Technical Conference*, Boston, MA, Jun. 2012.
- [21] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The Google file system," in *Proc. Nineteenth ACM Symposium on Operating Systems Principles*, Bolton Landing, NY, Oct. 2003.
- [22] D. Ford, F. Labelle, F. I. Popovici, M. Stokely, V.-A. Truong, L. Barroso, C. Grimes, and S. Quinlan, "Availability in globally distributed storage systems," in *Proc. 9th USENIX Symposium on Operating Systems Design and Implementation*, Vancouver, BC, CA, Oct. 2010.
- [23] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai, "Batch codes and their applications," in *Proc. Thirty-sixth Annual ACM Symposium on Theory of Computing*, Chicago, IL, Jun. 2004.

- [24] K. V. Rashmi, N. B. Shah, and P. V. Kumar, “Enabling node repair in any erasure code for distributed storage,” in *Proc. 2011 IEEE International Symposium on Information Theory (ISIT 2011)*, Saint Petersburg, Russia, Jul. 2011.
- [25] S. El Rouayheb and K. Ramchandran, “Fractional repetition codes for repair in distributed storage systems,” in *Proc. 48th Annual Allerton Conference on communication, control and computing*, Monticello, IL, Sep. 2010.
- [26] C. Tian, B. Sasidharan, V. Aggarwal, V. A. Vaishampayan, and P. Vijay Kumar, “Layered, exact-repair regenerating codes via embedded error correction and block designs,” *CoRR*, vol. abs/1408.0377, Aug. 2014.
- [27] O. Olmez and A. Ramamoorthy, “Fractional repetition codes with flexible repair from combinatorial designs,” *IEEE Trans. Inf. Theory*, vol. 62, no. 4, pp. 1565–1591, April 2016.
- [28] N. Silberstein and T. Etzion, “Optimal Fractional Repetition Codes Based on Graphs and Designs,” *IEEE Trans. Inf. Theory*, vol. 61, no. 8, pp. 4164–4180, Aug 2015.
- [29] G. M. Kamath, N. Prakash, V. Lalitha, and P. V. Kumar, “Codes With Local Regeneration and Erasure Correction,” *IEEE Trans. Inf. Theory*, vol. 60, no. 8, pp. 4637–4660, Aug 2014.
- [30] I. Tamo and A. Barg, “A Family of Optimal Locally Recoverable Codes,” *IEEE Trans. Inf. Theory*, vol. 60, no. 8, pp. 4661–4676, Aug 2014.
- [31] M. Blaum, J. Hafner, and S. Hetzler, “Partial-MDS codes and their application to RAID type of architectures,” *IEEE Trans. Inf. Theory*, vol. 59, no. 7, pp. 4510–4519, July 2013.
- [32] M. Blaum and J. S. Plank, “Construction of two SD codes,” *CoRR*, vol. abs/1305.1221, May 2013.
- [33] B. Gaston, J. Pujol, and M. Villanueva, “A realistic distributed storage system: the rack model,” *CoRR*, vol. abs/1302.5657, Feb. 2013.
- [34] N. Prakash, V. Abdrashitov, and M. Médard, “The storage vs repair-bandwidth trade-off for clustered storage systems,” *CoRR*, vol. abs/1701.04909, Jan. 2017.
- [35] F. J. MacWilliams and N. J. A. Sloane, *The theory for error-correcting codes*. North-Holland, 1977.
- [36] N. Silberstein, A. S. Rawat, and S. Vishwanath, “Error resilience in distributed storage via rank-metric codes,” in *Proc. 50th Annual Allerton Conference on communication, control and computing*, Monticello, IL, Oct. 2012.
- [37] —, “Error-Correcting Regenerating and Locally Repairable Codes via Rank-Metric Codes,” *IEEE Trans. Inf. Theory*, vol. 61, no. 11, pp. 5765–5778, Nov 2015.
- [38] O. Koyluoglu, A. Rawat, and S. Vishwanath, “Secure cooperative regenerating codes for distributed storage systems,” *IEEE Trans. Inf. Theory*, vol. 60, no. 9, pp. 5228–5244, Sep. 2014.
- [39] G. Calis and O. O. Koyluoglu, “A general construction for pmfs codes,” *IEEE Communications Letters*, vol. PP, no. 99, pp. 1–4, Nov. 2016.
- [40] T. Ho, M. Medard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, “A random linear network coding approach to multicast,” *IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.
- [41] E. M. Gabidulin, “Theory of codes with maximum rank distance,” *Problemy Peredachi Informatsii*, vol. 21, no. 1, pp. 3–16, 1985.
- [42] G. Calis and O. O. Koyluoglu, “Repairable block failure resilient codes,” in *Proc. 2014 IEEE International Symposium on Information Theory (ISIT 2014)*, Honolulu, HI, Jul. 2014.
- [43] D. R. Stinson, *Combinatorial designs: construction and analysis*. Springer, 2004.

APPENDIX A

CONCATENATED GABIDULIN AND MDS CODING

Set $K = (b - \rho)k_c$, and consider data symbols $\{u_0, \dots, u_{K-1}\}$.

- Use $[N = K + \rho k_c, K, D]$ Gabidulin code to encode $\{u_0, \dots, u_{K-1}\}$ to length- N codeword (x_1, \dots, x_N) . That is $(x_1, \dots, x_N) = (f(g_1), \dots, f(g_N))$, where the linearized polynomial $f(g) = u_0 g^{[0]} + \dots + u_{K-1} g^{[K-1]}$ is constructed with N linearly independent, over \mathbb{F}_q , generator elements $\{g_1, \dots, g_N\}$ each in \mathbb{F}_{q^M} ; and its coefficients are selected by the length- K input vector. We represent this operation by writing $\mathbf{x} = \mathbf{u} \mathbf{G}_{\text{MRD}}$.
- Split resulting N symbols $\{x_1, \dots, x_N\}$ into b blocks each with k_c symbols. We represent this operation by double indexing the codeword symbols, i.e., $x_{i,j}$ is the symbol at block

```

1: Set  $\mathcal{B}_0^* = \emptyset$  and  $i = 1$ 
2: while  $H(\mathbf{c}(\mathcal{B}_{i-1}^*)) < \mathcal{M}$  do
3:   Pick a coded local group  $\mathbf{c}_i \notin \mathcal{B}_{i-1}^*$  s.t.  $|\mathcal{B}_i \setminus \mathcal{B}_{i-1}^*| \geq \rho_L$ 
4:   if  $H(\mathbf{c}(\mathcal{B}_{i-1}^*), \mathbf{c}(\mathcal{B}_i)) < \mathcal{M}$  then
5:     set  $\mathcal{B}_i^* = \mathcal{B}_{i-1}^* \cup \mathcal{B}_i$ 
6:   else if  $H(\mathbf{c}(\mathcal{B}_{i-1}^*), \mathbf{c}(\mathcal{B}_i)) \geq \mathcal{M}$  and  $\exists \mathcal{B}'_i$  s.t.  $\mathcal{B}'_i = \arg \max_{\mathcal{B}'_i \subset \mathcal{B}_i} H(\mathbf{c}(\mathcal{B}_{i-1}^*), \mathbf{c}(\mathcal{B}'_i)) < \mathcal{M}$  then
7:     set  $\mathcal{B}_i^* = \mathcal{B}_{i-1}^* \cup \mathcal{B}'_i$ 
8:   end if
9:    $i = i + 1$ 
10: end while
11: Output:  $\mathcal{B}^* = \mathcal{B}_{i-1}^*$ 

```

Fig. 11: Construction of a set \mathcal{B}^* with $H(\mathbf{c}(\mathcal{B}^*)) < \mathcal{M}$ for BFR-LRC

i and j for $i = 1, \dots, b$, $j = 1, \dots, k_c$. We also denote the resulting sets with the vector notation, $\mathbf{x}_{i,1:k_c} = (x_{i,1}, x_{i,2}, \dots, x_{i,k_c})$ for block i .

- Use an $[n = c, k = k_c, d]$ MDS array code for each block to construct additional parities. Representing the output symbols as $\mathbf{y}_{i,1:c}$ we have $\mathbf{y}_{i,1:c} = \mathbf{x}_{i,1:k_c} G_{\text{MDS}}$ for each block i , where G_{MDS} is the encoding matrix of the MDS code over \mathbb{F}_q . For instance, if a systematic code is used, $\mathbf{x}_{i,1:k_c}$ is encoded into the vector $\mathbf{y}_{i,1:n} = (x_{i,1}, \dots, x_{i,k_c}, p_{i,1}, \dots, p_{i,c-k_c})$ for each block $i = 1, \dots, b$.

In the resulting code above if one erases ρ blocks and any $c - k_c$ symbols from the remaining blocks, the remaining $(b - \rho)k_c$ symbols form linearly independent evaluations of the underlying linearized polynomial which can be decoded due to the Gabidulin code from which the data symbols can be recovered and hence, by re-encoding, the pre-erasure of the version of the system can be recovered.

APPENDIX B

PROOF OF THEOREM 34

Proof. In order to get an upper bound on the resilience of an BFR-LRC, the definition of resilience given in (28) is utilized similar to proof in [9], [10]. We iteratively construct a set $\mathcal{B}^* \subset \mathcal{B}$ so that $H(\mathbf{c}(\mathcal{B}^*)) < \mathcal{M}$. The algorithm is presented in Fig. 11. Let b_i and h_i represent the number of blocks and entropy included at the end of the i -th iteration. We define the following:

$$b_i = |\mathcal{B}_i^*| - |\mathcal{B}_{i-1}^*| \leq b_L, \quad (49)$$

$$h_i = H(\mathbf{c}(\mathcal{B}_i^*)) - H(\mathbf{c}(\mathcal{B}_{i-1}^*)) \leq K_L. \quad (50)$$

Assume that the algorithm outputs at $(l+1)^{th}$ iteration then it follows from (49) and (50) that⁴⁰

$$|\mathcal{B}^*| = |\mathcal{B}_l^*| = \sum_{i=1}^l b_i \quad H(\mathbf{c}(\mathcal{B}^*)) = H(\mathbf{c}(\mathcal{B}_l^*)) = \sum_{i=1}^l h_i. \quad (51)$$

The analysis of algorithm is divided into two cases as follows.

- Case 1: [Assume that the algorithm exits without ever entering line 7.] We have

$$h_i = H(\mathbf{c}(\mathcal{B}_i^*)) - H(\mathbf{c}(\mathcal{B}_{i-1}^*)) = H(\mathbf{c}(\mathcal{B}_i^* \setminus \mathcal{B}_{i-1}^*) | \mathbf{c}(\mathcal{B}_{i-1}^*)) \leq (b_i - \rho_L) \frac{K_L}{b_L - \rho_L}. \quad (52)$$

From which we can obtain $b_i \geq \frac{h_i(b_L - \rho_L)}{K_L} + \rho_L$. Then, we derive the following:

$$|\mathcal{B}^*| = |\mathcal{B}_l^*| = \sum_{i=1}^l b_i \geq \sum_{i=1}^l \left(\frac{h_i(b_L - \rho_L)}{K_L} + \rho_L \right) = \frac{b_L - \rho_L}{K_L} \sum_{i=1}^l h_i + \rho_L l. \quad (53)$$

Similar to proof in [10], we have

$$\sum_{i=1}^l h_i = \left\lceil \frac{\mathcal{M}}{K_L/(b_L - \rho_L)} \right\rceil \frac{K_L}{b_L - \rho_L} - \frac{K_L}{b_L - \rho_L} \quad l = \left\lceil \frac{\mathcal{M}}{K_L} \right\rceil - 1. \quad (54)$$

By combining (53) with the above, we obtain

$$|\mathcal{B}^*| \geq \left\lceil \frac{\mathcal{M}(b_L - \rho_L)}{K_L} \right\rceil - 1 + \rho_L \left(\left\lceil \frac{\mathcal{M}}{K_L} \right\rceil - 1 \right). \quad (55)$$

- Case 2: [The algorithm exits after entering line 7.] For this case, we have

$$H(\mathbf{c}(\mathcal{B}_{i-1}^*), \mathbf{c}(\mathcal{B}_i)) \geq \mathcal{M}. \quad (56)$$

In each iteration step we add K_L entropy, hence we have

$$l \geq \left\lceil \frac{\mathcal{M}}{K_L} \right\rceil. \quad (57)$$

For $i \leq l-1$, same as before, we have $b_i \geq \frac{h_i(b_L - \rho_L)}{K_L} + \rho_L$. For $i = l$, we have $b_l \geq \frac{h_l}{K_L/(b_L - \rho_L)}$. Hence, it follows from (51) and (57) that

$$\begin{aligned} |\mathcal{B}^*| &= \sum_{i=1}^l b_i \geq \sum_{i=1}^{l-1} \left(\frac{h_i(b_L - \rho_L)}{K_L} + \rho_L \right) + \frac{h_l}{K_L/(b_L - \rho_L)} = \frac{b_L - \rho_L}{K_L} \sum_{i=1}^l h_i + (l-1)\rho_L \\ &\geq \frac{b_L - \rho_L}{K_L} \left(\left\lceil \frac{\mathcal{M}(b_L - \rho_L)}{K_L} \right\rceil \frac{K_L}{b_L - \rho_L} - \frac{K_L}{b_L - \rho_L} \right) + \left(\left\lceil \frac{\mathcal{M}}{K_L} \right\rceil - 1 \right) \rho_L \\ &= \left\lceil \frac{\mathcal{M}(b_L - \rho_L)}{K_L} \right\rceil - 1 + \left(\left\lceil \frac{\mathcal{M}}{K_L} \right\rceil - 1 \right) \rho_L. \end{aligned} \quad (58)$$

Therefore, by combining (28), (55) and (58) we have

$$\rho \leq b - \left\lceil \frac{\mathcal{M}(b_L - \rho_L)}{K_L} \right\rceil - \left(\left\lceil \frac{\mathcal{M}}{K_L} \right\rceil - 1 \right) \rho_L. \quad (59)$$

□